December 1977

This document describes the differences between RSTS/E releases and establishes a maintenance notebook for documentation corrections, published software problems and solutions, and programming notes.

RSTS/E V06C
Release Notes

Order No. AA-5246B-TC

SUPERSESSION/UPDATE INFORMATION:	This contains information concerning RSTS/E V06C-03 and replaces the previous document titled RSTS/E V06B Release Notes, Order No. DEC-11-ORINC-A-D and updates -DN1, -DN2, and -DN3.
OPERATING SYSTEM AND VERSION:	RSTS/E V06C-03
SOFTWARE VERSION:	RSTS/E V06C-03

To order additional copies of this document, contact the Software Distribution Center, Digital Equipment Corporation, Maynard, Massachusetts 01754

digital equipment corporation • maynard. massachusetts

First Printing, December 1977

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

Copyright © 1977 by Digital Equipment Corporation

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-11
DECCOMM	DECSYSTEM-20	TMS-11
ASSIST-11	RTS-8	ITPS-10

10/78-15

CONTENTS

1.0	INTRODUCTION
2.0	DIFFERENCES BETWEEN RSTS/E V06C-03 AND V06B-02
2.1	New Processor Support
2.2	New Device Support
2.2.1	RM03
2.2.2	Magnetic Tape Support
2.2.3	ECC MOS Memory
2.2.4	DMC-11 Interprocessor Link
2.3	System Generation
2.4	Initialization Code
2.5	Monitor and File Processing
2.5.1	Job Spawning Executive SYS Call
2.5.2	Hangup a Dial-up Line SYS Call -- Additional Functionality
2.5.3	SLEEP Function -- Additional Functionality
2.5.4	New File OPEN Modes
2.6	Terminal Service
2.6.1	New Terminal OPEN Modes
2.6.2	Echo Control
2.6.3	RETYPE Function
2.6.4	XOFF/XON Protocol
2.6.5	Terminal Characteristics
2.6.6	Pseudo Keyboard Processing
2.7	BASIC-PLUS Language and Run-Time System
2.8	RSTS/E Utility Library Programs
2.8.1	RSTS/E System Utilities Package
2.8.2	RSTS/E Data Manipulation Package
2.8.3	RSTS/E Error Control Package
2.8.4	RSTS/E Backup Package
2.8.5	RSTS/E Spooling and Operator Services Package
2.8.6	RSTS/E Run-Time System Support Package
2.8.7	RSTS/E User Environment Test Package
2.8.8	RSTS/E Automated Patching Facility Package
2.9	RSX Utilities Package
2.9.1	RSX Run-Time System
2.9.2	RSX Utilities
2.10	RMS-11 Software Package
2.11	RT-11 Utilities Package
2.11.1	PIP.SAV
2.11.2	ONLPAT.SAV
2.12	Documentation
2.13	TECO and RNO
2.14	MACRO Programming
2.15	Distribution Media
2.16	Optional Software Supported
2.17	Old Problems Corrected
2.17.1	RSTS/E Monitor
2.17.2	BASIC-PLUS Language and Run-Time System
2.17.3	Commonly Used System Programs (CUSPs)
2.17.4	Run-Time Systems and Support
2.18	Upgrading from V06B-02 to V06C-03

3.0 CONTENTS OF RSTS/E V06C-03 DISTRIBUTION KITS
4.0 NOTEBOOK MAINTENANCE
4.1 RSTS/E COMPONENT. SUB-COMPONENT Assignments
4.2 Subordinate Software
5.0 SOFTWARE PERFORMANCE REPORTS
5.1 Software Performance Report Guidelines
6.0 PATCHING PROCEDURES
6.1 Monitor Patching
6.2 Commonly Used System Program (CUSP) Patching
7.0 RELEASE NOTES AND SOFTWARE DISPATCH ARTICLES
7.1 Cumulative Index of all Articles
7.2 Commonly Used System Programs (CUSPs) Status Report
7.3 Commonly Used System Programs (CUSPs) Protection Codes
8.0 SOFTWARE PRODUCT DESCRIPTIONS (SPDs)
8.1 Software Product Descriptions (SPDs)
8.2 Option Bulletins

APPENDIX A SYSGEN EDITOR MANUAL

1.Ø Introduction

The RSTS/E VØ6C-Ø3 Release Notes establish a maintenance notebook for documentation corrections, published software problems and solutions, and programming notes. Subordinate software information is also included. The Software Dispatch is designed to be inserted into the maintenance notebook. It is provided for one year after each Category A or B software installation, and is addressed to the software contact established by the DIGITAL field office. After one year, the customer may purchase the Binary Program Update Service to continue receiving the Dispatch.

Anyone who is about to generate RSTS/E VØ6C-Ø3 must incorporate "Mandatory" and appropriate "Optional" patches as part of the system generation procedure. The patches are summarized in the Cumulative Index. There is also an index of the version or edit levels for all of the Commonly Used System Programs (CUSPs). These indices are updated regularly in the Software Dispatch.

For further information on RSTS/E documentation, refer to the RSTS/E Documentation Directory.

2.Ø Differences Between RSTS/E VØ6C-Ø3 and VØ6B-Ø2

RSTS/E VØ6C-Ø3 introduces new hardware support and several new software features.

All published patches for VØ6B-Ø2 through the December 1977 Software Dispatch have been incorporated.

Several of the VØ6B-Ø2 manuals are updated to reflect new hardware support and new software features. In addition, several new manuals are incorporated in the RSTS/E manual set.

Section 2.18 presents guidelines for installations upgrading from VØ6B-Ø2 to VØ6C-Ø3.

2.1 New Processor Support

The PDP 11/6Ø processor is now supported. Features include 11/6Ø error logging, cache control and error recovery.

2.2 New Device Support

Several devices introduced since VØ6B-Ø2 was released are supported in VØ6C-Ø3.

2.2.1 RMØ3

The RMØ3 disk is a high speed, high capacity disk using the name DR:. An RMØ3 disk pack can hold 67 million bytes. An RMØ3 disk may be used as the system disk, a public disk, or a private disk. Like the RPØ4/Ø5/Ø6 disks, the RMØ3 disk uses the RH7Ø interface to the 11/7Ø processor. However, when both types of disks are included in a system, there must be one RH7Ø for the RMØ3 disks, and a separate RH7Ø for the RP type disks. The RMØ3 is supported only on PDP-11/7Ø processors.

2.2.2 Magnetic Tape Support

RSTS/E VØ6C-Ø3 now supports the following 7 and 9-track tape drives:

TSØ3	9-track only	
TU1Ø	7 and 9-track	
TE1Ø	7 and 9-track	(replaces the TU1Ø)
TU16	9-track only	
TE16	9-track only	(replaces the TU16)
TU45	9-track only	

The new TE1Ø and TE16 drives are replacement products for the TU1Ø and TU16, respectively. (The TU1Ø and TU16 continue to be supported on existing RSTS/E systems.) On a given RSTS/E system, all TSØ3 and TU1Ø/TE1Ø tape drives form the MT device class, and all TU16/TE16 and TU45 tape drives form the MM device class. Up to eight units of each class may be used by RSTS/E VØ6C-Ø3.

2.2.3 ECC MOS memory

Error Correction Code (ECC) MOS memory is supported on the 11/6Ø and 11/7Ø processors. ECC MOS memory on the 11/7Ø is supported by RSTS/E software identically as parity memory and is in fact indistinguishable from parity memory. Single-bit parity errors in the memory are corrected by the ECC automatically and transparently. Double-bit errors are reported as parity errors.

2.2.4 DMC-11 Interprocessor Link

RSTS/E VØ6C-Ø3 can use up to 16 DMC-11 interprocessor links. The DMC-11 may be used for point-to-point communication between two machines.

2.3 System Generation

The System Generation procedure for VØ6C-Ø3 is similar to the VØ6B-Ø2 procedure.

The SYSGEN dialogue has several new questions which correspond to the new features of VØ6C-Ø3. The Automated Patching Facility of VØ6C-Ø3 can be included during the system generation to fix problems in the Monitor, the BASIC-PLUS Run-Time System, the RT11 Run-Time System, the SYSGEN Monitor, and the System Initialization code. The Monitor Statistics option, which was a "hidden feature" in previous releases, can now be selected in the SYSGEN process. This option is primarily intended for use by software provided by DIGITAL, such as the User Environment Test Package.

NOTE

DIGITAL assumes no responsibilty for the accuracy of data obtained by the statistics option, nor for the interpretation of that data. It is strongly recommended that you consult with DIGITAL representatives before undertaking any system monitoring operations.

The RT11.RTS and PIP.SAV files may be copied to a target system if the target disk is not the system disk. The BASIC-PLUS LOG and TRIG Functions can be included separately in the Run-Time System. The DATE and TIME format questions have been moved to the DEFAULT option of INIT.SYS.

Changes to existing questions include:

1) a minimum of one pseudo keyboard for all systems;

2) the number of DMC11'S may be Ø to 16;

3) 5 to 5Ø system-wide logical names;

4) some hardware-related questions are now specified at INIT time.

2.4 Initialization Code

To accommodate systems with non-standard hardware, the BOOT option enables only the keyboard console, disks and tapes. Only after using the HARDWR option for anything besides the LIST suboption, or after the first use of the INSTALL option, will INIT scan the entire hardware configuration. This change simplifies installation of RSTS/E on a number of non-standard hardware configurations that were difficult to handle in VØ6B-Ø2.

The new RMØ3 is a bootstrappable device and can be initialized by the DSKINT option.

Several changes were made to the DEFAULT option, including specifying DATE and TIME formats, the POWER FAIL delay, and the system time base clock. DEFAULT sets the initial SWAP MAX to 16K, allows an "ADDR+SIZE" construct for the Memory Table suboption, and detects ECC MOS memory.

The CLEAN operation of the REFRESH option is more rigorous. It detects and corrects more inconsistencies in the file structure, including bad directory links and links to empty entries.

The DSKINT option can indicate, on newly initialized disks, that new files are to be placed first in User File Directories; can retain bad block information when reinitializing RSTS/E disks; uses the FORMAT ENABLE switch with the DP formatter; and uses a CTRL/Z as a command to back up one question in the dialogue phase to correct typing errors.

The START option prints the Date and Time upon Auto-Restart.

The HARDWR option lists new data, including CPU options, and adds two new suboptions: 1) HERTZ, to specify the AC line frequency being used; and 2) SWITCH, to enable or disable the console switch register, or mark it as volatile.

The SET option allows the user to specify the line printer backspace characteristics.

The main initialization code has been improved to time out all loops, check the system clocks, disable cache memory if it is not working, handle cache errors during initialization, recognize RXØ1's installed on the system, and accept date information in either ALPHA or ISO format and time information in AM/PM or 24-hour format.

2.5 Monitor and File Processing

A few changes have been made to the monitor and file service of RSTS/E.

2.5.1 Job Spawning Executive SYS Call

There is a new JOB SPAWNING Executive call which allows a job to create another job without using a pseudo keyboard. This privileged call creates a new job, has it enter the System Default Run-Time System, and causes it to run a specified compiled program under any valid Run-Time System.

2.5.2 Hangup a Dial-up Line SYS Call -- Additional Functionality

The Hangup SYS call can enable carrier on a dial-up line, facilitating manual dial-out.

2.5.3 SLEEP Function -- Additional Functionality

The SLEEP function has been modified to prevent race conditions which were possible in VØ6B-Ø2. A SLEEP now awakens on any of the following conditions:

1) SLEEP time expires.

2) A delimiter is typed on any terminal ASSIGNed to or OPENed by the job.

3) A message becomes queued for the job.

4) Logins become disabled (e.g., using the UTILTY command NO LOGINS).

5) A state change occurs on a pseudo keyboard assigned to the job. (The job has PRINTed output for the controlling job to read, or has entered an input wait state.)

6) The XM: (DMC-11) driver is open and a message becomes pending for the job. This is a new condition for VØ6C-Ø3.

The SLEEP function now operates as follows:

SLEEP Ø%

Do not sleep, but reschedule job.

SLEEP N% where N%>Ø

SLEEP until one of the above conditions occurs (same as VØ6B-Ø2).

SLEEP N% + 32767% + 1% where Ø%<=N%<=32767%

If the operand is negative, the monitor checks to see whether any of the following conditions is true:

A) A delimiter has been typed on any terminal OPENed by the job, or a delimiter has been typed on any terminal ASSIGNed to this job and the console keyboard for this job is OPEN on some non-zero channel.

B) A message has been queued for the job.

C) A state change has occurred on a pseudo keyboard ASSIGNed to the job. (The job has PRINTed output for the controlling job to read, or has entered an input wait state.)

D) The XM: (DMC-11) driver is open and a message is pending for the job.

SLEEP does not occur if any of the conditions is pending. Otherwise, SLEEP continues until any of conditions 1) to 6) occurs. If N% is zero, the SLEEP time is infinite, and the job will awaken when one of the other conditions occurs.

2.5.4 New File OPEN Modes

As noted in section 2.4, disks may be initialized so that new files are always placed first in file directories. Two new OPEN modes can be used to override the default placement:

1) MODE 1024% creates a new file at the end of the file directory.

2) MODE 1536% (1024% + 512%) creates a new file as the first file in the directory.

Placing files first in a directory speeds up access to these files. Note that directories should be reordered more often to minimize the chance of directory links crossing blocks.

2.6 Terminal Service

2.6.1 New Terminal OPEN Modes

There are three new terminal OPEN modes:

16 Guarded terminal mode

This mode will resturn error 27 (rather than hibernating) if I/O is done when detached. This allows programs to check for dial-up lines becoming disconnected. Also, CTRL/C wipes output but doesn't abort the job. The next GET from the terminal will return ASCII(3%) to signal the CTRL/C.

32 Incoming XOFF/XON protocol mode

Used with MODE 1% (binary input mode), this mode passes all incoming characters to the program except for XOFF (CTRL/S) and XON (CTRL/Q), which have their normal meanings (XOFF suspends output, and XON resumes output). Used without MODE 1%, this mode recognizes only XOFF and XON. This mode is used by the line printer spooler when printing on LA180S terminals.

128 Special scope rubout mode

If there is a character to rubout, and the character is a normal graphic (CHR$(32) to CHR$(126)), then a normal scope rubout is performed.

If there is no character to rubout, or the character is a non-graphic (CHR$(Ø) to CHR$(31) or CHR$(127)), rubout is buffered as a delimiter and the job is scheduled. This mode is used by TECO and should be useful for other scope-oriented editors.

NOTE

MODE 128% is reserved for use by software supplied by Digital, and is subject to change in future releases.

2.6.2 Echo Control

Echo Control (MODE 8%) has been extended. In VØ6B-Ø2, the "Declare a Field" operation accepted a one or two byte record: the first byte defined the field length and the second byte the "paint" character. Any additional bytes were ignored. In VØ6C-Ø3, this operation is extended as follows:

If the COUNT value is one or two, the operation is identical to that of VØ6B-Ø2.

If the COUNT value is greater than two, PRINT the first N-2 bytes on the terminal as a prompt, then define a field whose length is defined by byte N-1, and whose "paint" character is defined by byte N.

2.6.3 RETYPE Function

Typing CTRL/R causes all buffered terminal input, including type-ahead input, to be printed out. This function is particularly useful on non-scope terminals. After typing a line which has characters deleted by the RUBOUT or DELETE key, and before typing a line delimiter, typing a CTRL/R will cause the system to retype the entire line with the deleted characters removed, allowing the user to inspect the exact text to be transmitted. This mode may be enabled or disabled by use of the TTYSET SYS function call.

2.6.4 XOFF/XON Protocol

The XOFF/XON (CTRL/S, CTRL/Q) feature has been modified. After output has been suspended by typing CTRL/S, the output will be resumed only by typing CTRL/Q. While output is suspended, typed characters are buffered but not echoed. The system echoes the buffered characters when terminal output is resumed. This follows the DEC standard for terminal synchronization. In VØ6B-Ø2 and previous versions, typing any character other than CTRL/S resumed output. The TTYSET SYS function call can specify the VØ6B-compatible "resume on any character" mode.

2.6.5 Terminal Characteristics

The TTYSET SYS function call returns the current terminal characteristics. It allows non-privileged users to set their own terminal characteristics. Privilege is required only to set characteristics of other terminals, or to specify the /RING switch.

2.6.6 Pseudo Keyboard Processing

Pseudo keyboard processing has changed as follows: If the PK: unit is opened MODE Ø%, the controlled job will be KILLed if the PK: unit is closed. If the PK: unit is opened MODE 1%, the controlled job will be detached if the PK: unit is closed. This change from VØ6B-Ø2 was made to improve system security by making it more difficult for a non-privileged user to detach a job. An optional patch allows the system manager to make MODE 1% privileged.

An additional PUT RECORD mode is available: PUT #N%, RECORD 16% kills any job currently running on the controlled KB:.

The TTYSET SYS function can set default "terminal" characteristics for controlled keyboards by using the /RING switch.

2.7 BASIC-PLUS Language and Run-Time System

There are new keywords in BASIC-PLUS for full RECORD I/O capability.

The GET and PUT statements can specify:

1) BLOCK <expression>, to specify block number, where <expression> can be either an integer or floating-point expression;

2) COUNT <limit>, to specify a limiting count for GET or PUT; and

3) USING <offset>, to specify an offset into the buffer.

NOTE

These keywords are not currently implemented in BASIC-PLUS-2.

A new SYS call (SYS(CHR$(12%)) returns the contents of the File Request Block (FIRQB) in the format used by the code 6 SYS calls. This call can be used:

1) After a program starts, to determine the source (i.e., device, PPN, filename, etc.) of the program;

2) After an OPEN, to determine what file was OPENed.

NOTE

SYS(CHR$(12%)) is available only under BASIC-PLUS, not BASIC-PLUS-2.

The ATTACH command has been deleted from BASIC-PLUS. The CCL ATTACH command must be installed.

2.8 RSTS/E Utility Library Programs

Rather than describe the entire set of Commonly Used System Programs (CUSPs) as a single library, the programs are grouped into packages which are included in each RSTS/E Distribution kit. The packages are:

RSTS/E System Utilities Package
RSTS/E Data Manipulation Package
RSTS/E Error Control Package
RSTS/E Backup Package
RSTS/E Spooling and Operator Services Package
RSTS/E Run-Time System Support Package
RSTS/E User Environment Test Package
RSTS/E Automated Patching Facility Package

The changes in programs in each of these packages are described below, along with brief descriptions of new programs being introduced in VØ6C-Ø3.

2.8.1 RSTS/E System Utilities Package

The RSTS/E System Utilities Package contains the following:

BUILD	DIRECT	DISPLY	DSKINT	GRIPE
INIT	INUSE	LOGIN	LOGOUT	MONEY
ODT	PLEASE	PRIOR	QUOLST	REACT
REORDR	SHUTUP	SWITCH	SYSCAT	SYSTAT
TALK	TTYSET	UMOUNT	UTILTY	UTILT1

BUILD

The BUILD program now uses logical device names instead of the unusual "?:" construct to determine the destination of the built files. The "?:" construct still works, but the control files distributed with RSTS/E VØ6C-Ø3 use logical device names. In a BUILD to a non-system disk, no files are built onto the system disk. The questions in a System Build have been reorganized. CTRL/C can be typed on the controlled keyboard to abort the BUILD run.

DIRECT

New switches have been added to DIRECT to display more information about the current status of each file. DECtape and magtape directories are now handled by DIRECT, rather than by CHAINing to $PIP.

DISPLY (VTØ5, VT5Ø(VT52))

Most of the new features in SYSTAT have been added to the dynamic display programs. The DISPLY base program flags Non-Swappable jobs, disks initialized with the New Files First option, and busy devices owned by NSP (DECnet Network Services Protocol). Maximum Run-Time System job size was not added; it could not fit on the screen.

The VTØ5 and VT5Ø control modules function in VØ6C-Ø3 as they did in VØ6B-Ø2.

DSKINT

DSKINT allows the user to specify whether new directory entries are to be added to the beginning or to the end of the directory (New Files First option).

In VØ6B-Ø2, DSKINT always used the information contained in the old bad block file (if the disk was RSTS/E file structured) when it built the new bad block file. This is now optional. (It might be desirable to ignore the old bad block information if, for example, the disk is known to be corrupt.)

GRIPE

The GRIPE program functions in VØ6C-Ø3 as it did in VØ6B-Ø2.

INIT

The INIT program can be aborted by typing CTRL/C on any controlled keyboard. Note that the CTRL/C will be detected the next time INIT attempts to FORCE to that keyboard.

INUSE

The INUSE program now runs non-privileged.

LOGIN

The LOGIN program functions in VØ6C-Ø3 as it did in VØ6B-Ø2.

LOGOUT

LOGOUT does not allow users to delete files that are write-protected against them.

MONEY

The MONEY program functions in VØ6C-Ø3 as it did in VØ6B-Ø2.

ODT

The ODT program functions in VØ6C-Ø3 as it did in VØ6B-Ø2.

PLEASE

The PLEASE program functions in V06C-03 as it did in V06B-02.

PRIOR

The PRIOR program functions in V06C-03 as it did in V06B-02.

QUOLST

The QUOLST program functions in V06C-03 as it did in V06B-02.

REACT

The REACT program functions in V06C-03 as it did in V06B-02.

REORDR

The REORDR program functions in V06C-03 as it did in V06B-02.

SHUTUP

Unsolicited input typed on the system console will not cause SHUTUP to fail. However, as in V06B-02, unsolicited input will not influence the speed with which the SHUTUP program brings the RSTS/E system down.

SWITCH

The SWITCH program is a new program for V06C-03 which allows a user to change his default Run-Time System.

SYSCAT

The SYSCAT program functions in V06C-03 as it did in V06B-02.

SYSTAT

SYSTAT lists the following new data:

Section	Additions
Job Status	Indication of jobs which are Non-Swappable.
Disk Structure	Indication of disks initialized with the New Files First option.
Buffer Status	Indication of devices owned by NSP (DECnet Network Services Protocol).
Run-Time Systems	Maximum job size for the RTS.

TALK

The TALK program functions in V06C-03 as it did in V06B-02.

TTYSET

TTYSET is faster and more efficient. The implied SPEED settings in TTYSET's MACROs have been removed. (In V06B, problems arose if the TTYSET.SPD file did not permit the SPEED included in the appropriate MACRO on the desired terminal.) The TTYSET.SPD file is no longer required. If TTYSET does not find the TTYSET.SPD file, or finds that a particular terminal does not have an entry in TTYSET.SPD, it will default SPEED settings by interface type. The main purpose of TTYSET.SPD now is to limit the SPEED settings allowed for specific terminal lines.

The following new TTYSET options are available:

LIST	List current terminal characteristics.
CTRL/R	Enable/Disable the CTRL/R Retype facility.
RESUME	Define XOFF/XON processing: "RESUME ANY" enables the V06B-02 mode whereby any character typed after XOFF resumes typeout and echo; while "RESUME CTRL/C" resumes typeout only after typing XON or CTRL/C.
/RING	May be used by privileged users to specify characteristics of disabled terminals. This allows setting initial characteristics (such as WIDTH) for pseudo keyboards).

All these features are explained in the RSTS/E System User's Guide in the section on the TTYSET program.

UMOUNT

The UMOUNT program has new features to control magtape. In addition to assigning a magtape unit to the current or specified job, the user can specify the density and/or parity of the tape being mounted, and if the tape is to be mounted read-only. In addition, the volume identification label is expected (following the device designator) and checked when mounting an ANSI standard magtape. A warning message is printed if it is omitted.

UTILTY
UTILT1

The FORCE command in UTILTY can force control characters. It now operates exactly like the FORCE command in INIT.

The UTILT1 program functions in V06C-03 as it did in V06B-02.

2.8.2 RSTS/E Data Manipulation Package

The RSTS/E Data Manipulation Package contains the following:

BPCREF	BPCRF1	COPY	EDIT	EDITCH
FILCOM	FLINT	FLX	MAKRTS	PIPSML
PIPEXT	PMDUMP	RUNOFF		

BPCREF
BPCRF1

The BPCREF package produces a cross-reference listing of a compiled (.BAC) BASIC-PLUS program. The listing provides a list of all referenced line numbers and an alphabetized list of all variables used in the program and all line numbers at which those variables are referenced. Destructive references are specially flagged, as are the lines at which arrays and functions are defined. A separate list of line numbers which are referenced but not present, and variables which are set but never read or read but never set is included at the end of the listing. These and other features are described in the RSTS/E System User's Guide. Note that BPCREF works on BASIC-PLUS programs only.

COPY

The COPY program functions in VØ6C-Ø3 as it did in VØ6B-Ø2.

EDIT
EDITCH

The EDIT/EDITCH package issues a warning or fatal message if a user attempts to edit a file with attributes. If an input file has attributes which indicate ASCII stream records, a warning message is issued and the editor proceeds. If an input file has attributes indicating any other record type, a fatal error message is issued and the run aborts. EDIT can process only ASCII stream files. The new EDT program (section 2.9.2) or TECO (section 2.13) may be used to edit files with attributes.

FILCOM

The FILCOM program issues a fatal error message if either of the input files has non-ASCII stream attributes.

FLINT

The FLINT program functions in VØ6C-Ø3 as it did in VØ6B-Ø2.

FLX

The FLX program functions in VØ6C-Ø3 as it did in VØ6B-Ø2.

MAKRTS

The MAKRTS program is used to convert DIGITAL-supplied Run-Time Systems built using the task builder (TKB.TSK) into the .RTS format expected by RSTS/E. Instructions for using MAKRTS are included in the program source. This utility is intended as a software maintenance tool.

PIPSML

The BASIC-PLUS program PIP.BAS has been superseded by the PIP.SAV program (see section 2.11.1). For small RSTS/E systems, and for those systems where some users may prefer to use the old PIP.BAS program, the program PIPSML.BAS (PIP-Small) has been included in the distribution kit.

PIPSML.BAS is similar to the old PIP.BAS program, except that the default transfer mode has been changed to normal copy mode ("/CO").

To specify that the CCL command "PIP" should run the program PIPSML, use the following command to UTILTY:

```
CCL PIP=SY:$PIPSML.BAC;30000
```

PIPEXT

The BASIC-PLUS program PIPEXT.BAS has been superseded by the PIP.SAV program (see section 2.11.1). For those systems that do not choose to make the RT11 Run-Time System generally available, and for those systems where some users may prefer it, the PIPEXT.BAS program has been included in the distribution kit.

PIPEXT.BAS has been enhanced for this release, as follows:

The default transfer mode has been changed to normal copy mode ("/CO").

Run-Time System names and file attributes are copied in disk to disk transfers.

Filename string scan switches, such as /MODE, are recognized, if they immediately follow the appropriate file name. (Filename string scan switches must precede any PIPEXT switches.)

To specify that the CCL command "PIP" should run the program PIPEXT, use the following command to UTILTY:

```
CCL PIP=SY:$PIPEXT.BAC;30000
```

PMDUMP

This new program is provided as a debugging tool for programs which run under the RSX Run-Time System or any of its derivatives. It provides a formatted listing from an RSX post-mortem dump file.

RUNOFF

The RUNOFF program functions in V06C-03 as it did in V06B-02.

2.8.3 RSTS/E Error Control Package

The RSTS/E Error Control Package contains the following programs:

ANALYS ANALY1
ERRINT ERRCPY ERRDIS ERRDET ERRBLD

ANALYS
ANALY1

The crash analysis programs incorporate all new features added to SYSTAT. In addition, ANALY1 now chains to ERRDIS to append a detailed report of the information in the newly created crash error logging file ($ERRCRS.FIL) to the regular crash dump report.

ERRINT
ERRCPY
ERRDIS
ERRDET
ERRBLD

The Error Control Package reports errors occurring on new RSTS/E supported devices. It takes advantage of expanded monitor error detection/logging capabilities.

The ERRBLD program has been added to the package. A list of changes to the programs in the package is presented below:

Program	Description
ERRINT	Functions in V06C-03 as it did in V06B-02.
ERRCPY	Functions in V06C-03 as it did in V06B-02.
ERRDIS	Handles all new error codes. Also, may now be CHAINed to from ANALY1 to automatically produce a detailed crash error logging file report.
ERRDET	Handles all new error codes; reports the new information supplied in monitor error logging records.
ERRBLD	Creates $ERRDAT.FIL, which contains specific error analysis information, such as error mnemonics, related registers and register bit descriptors. The information stored in this file includes all new ERRDET heading/device data; it is NOT the same as in the version of ERRDAT.FIL distributed with V06B-02. Since this new data file must be present if correct reports are to be generated, ERRBLD must be run to recreate ERRDAT.FIL prior to running ERRDIS for the first time (This is done automatically by the standard system library BUILD procedures.)

The Error Control Package provides a valuable diagnostic tool in the analysis of hardware and system software problems. Each installation is encouraged to read the RSTS/E System Manager's Guide for further information on the package.

2.8.4 RSTS/E Backup Package

The Backup Package supports the new disks and tapes. Also, the BACKUP volume mount program (BACMNT.BAS) includes a new SCRATCH option which enables a privileged user to specify that a magtape being mounted for a Backup operation should not undergo any tape label checking.

2.8.5 RSTS/E Spooling and Operator Services Package

The RSTS/E Spooling and Operator Services Package contains the following programs:

OPSER	OPSRUN	QUE	QUEMAN	QUMRUN
CHARS	SPOOL	SPLIDL	SPLRUN	
BATDCD	BATCH	BATIDL	BATDEC	BATRUN

OPSER
OPSRUN

The OPSRUN program allows any privileged user on KBØ: to use OPSER under the SHUTUP program to bring the system down. In VØ6B-Ø2, this would fail if the privileged user was not also in OPSER's valid operator table.

During system shutdown under the SHUTUP program, OPSER will forcibly take control of KBØ: if it cannot re-attach to KBØ:. This may happen if unsolicited input had been typed on KBØ:. When OPSER forcibly takes over KBØ:, CTRL/C characters may be displayed on the terminal.

QUE

The job queueing program, QUE, allows an account specification in the Jobname field of all QUE commands, except the KILL command. The JOB will run under the account specified. The privilege to include an account specification different from the current user's has been extended to all privileged users and all valid operators. If an account specification is specified which is different from the current user's, then the user is restricted to the level of privilege of the account specified for the duration of the command. The default account specification in the file specification is still the current user's account.

The TYPE:CBL switch has been removed; it was not needed to process COBOL files.

QUEMAN
QUMRUN

The QUEMAN program functions in VØ6C-Ø3 as it did in VØ6B-Ø2.

The QUMRUN program will print a message on the Operator's Services Console whenever an LP: queue has no further eligible JOBs to process and there are eligible JOBs waiting with a FORMNAME different than the current FORMNAME of the LP: Spooler. The LP*: queue is considered an extension of all the LP: queues, and, as such, a JOB queued to the LP*: queue may result in the "FORMNAME different" message for any of the LP: queues. It may be necessary to check the LP*: queue as well as the specific LP: queue to determine which JOB is responsible for the message.

NOTE

This feature may be disabled with QUMRUN Optional patch Seq. 14.6.1 0

CHARS
SPOOL
SPLIDL
SPLRUN

The Spooling Package functions in VØ6C-Ø3 as it did in VØ6B-Ø2.

BATDCD
BATCH
BATIDL
BATDEC
BATRUN

BATCH and BATIDL function in VØ6C-Ø3 as they did in VØ6B-Ø2.

BATDEC, the BATCH decoding program, has been modified as follows. The $COBOL command is now an invalid command and will return an error message to the log. It will also prevent execution of the BATCH job. This change was necessary because the requirements of COBOL/V3 changed drastically. COBOL may still be run under BATCH by using the $RUN command (i.e., "$RUN $COBOL").

The $FORTRAN and $BASIC commands produce all default files under the user's account. In VØ6B, the default files were created in the account under which the BATCH processor was running.

The $SORT command handles the following switches:

```
/ALL[OCATION]:n
/BLO[CKSIZE]:n
/BUC[KETSIZE]:n
/CON[TIGUOUS]
/DEV[ICE]:x
/FIL[ES]:n
/FOR[MAT]:x:n
/IND[EXEDSEQUENTIAL]:x
/KEY:abm.n
/PRO[CESS]:x
/REL[ATIVE]
/SEQ[UENTIAL]
/SIZ[E]:n
```

The /FILES and /FORMAT switches apply to the input file only. Refer to the PDP-11 Sort Reference Manual (AA-3341C-TC) for more information on switches.

The $FORTRAN command does not automatically link in $FORLIB.OBJ. In most RSTS/E VØ6C installations this is no longer necessary, because $FORLIB.OBJ has been incorporated into $SYSLIB.OBJ. The ability to specify additional libraries still exists. For additional information on $SYSLIB refer to the RSTS/E FORTRAN IV Installation Guide (AA-C762A-TC) and the RT11/RSTS/E FORTRAN IV User's Guide (DEC-11-LRRUB-A-D).

BATDCD is compatible with the new BATDEC. BATDCD must be run before any BATCH processing is done. (The BUILD procedures do this automatically when installing the Spooling programs.) BATDCD creates a special data file (BATCH.DCD) which BATDEC uses in the command decode operation. The BATCH.DCD file created by BATDCD.BAS in VØ6B-Ø2 can not be used by the VØ6C-Ø3 BATCH package.

2.8.6 RSTS/E Run-Time System Support Package

Several components in the Run-Time System Support Package which were used by the obsolete versions of COBOL (V2) and RPG II (V7) have been removed from the RSTS/E distribution kit. These components are:

LNGBLD.BAS	TKB.BAS
RTSLIB.RTS	BIGTKB.TSK

NOTE

RSTS/E VØ6C-Ø3 does not support COBOL (V2) or RPG II (V7)

2.8.7 RSTS/E User Environment Test Package

The DREXER program has been added to the UETP Package to exercise RMØ3 disks. Several program names have been changed for compatibility with the rest of the package.

2.8.8 RSTS/E Automated Patching Facility Package

The RSTS/E Automated Patching Facility Package contains the following programs:

PATCPY PBUILD CPATCH AUTOED

PATCPY

PATCPY copies selected files from a specific device and account. It is used to copy patch control files from the media which patch particular programs or packages of programs. All the patch files for a particular package can be copied simply by typing the name of the package (or packages). Wild card specifications can be used to access the files directly rather than by package name.

PBUILD

PBUILD asks the user for the location of text files to patch; whether or not to compile the patched programs, and, if so, where to save them; and whether or not the patched sources should be saved. It then accepts a command file name and detaches from the keyboard. By forcing commands to the keyboard, it patches the programs by running CPATCH, and then compiles the programs via the BASIC-PLUS OLD and COMPILE commands.

CPATCH

CPATCH accepts the name of the file to be patched and the command and log file names, checks the names for validity, sets up a work file and then CHAINs to AUTOED to perform the patch. Although CPATCH may be run at the user's keyboard, it is normally run by the program PBUILD.

AUTOED

AUTOED edits ASCII text files. It OPENs a temporary work file created by the program CPATCH before CPATCH CHAINs to AUTOED. The work file contains a description of the file containing the editting command, the file to write a log of the editting operations, the file to edit, and where to write the resulting editted file. AUTOED uses a command language which is a subset of EDIT-11's command language.

2.9 RSX Utilities Package

2.9.1 RSX Run-Time System

A keyboard monitor has been added to this Run-Time System. It accepts the following commands:

```
HELLO   BYE       RUN
ASSIGN  DEASSIGN  REASSIGN
MOUNT   DISMOUNT  UNSAVE
```

Commands may be abbreviated to three characters. CCL commands are also handled exactly as in BASIC-PLUS.

A Post-Mortem Dump capability has been added. Any .TSK file built by TKB with the /PM switch will generate a PM Dump file if the program aborts for any reason, including typing CTRL/C. The PMDUMP program can format this file for printing.

2.9.2 RSX Utilities

The following utilities are in VØ6C-Ø3:

PAT.TSK

PAT merges an original .OBJ file with a correction .OBJ file to produce a patched .OBJ file. PAT is intended as a software maintenance tool.

EDT.TSK

EDT can edit normal RSTS/E ASCII stream files or RMS-compatible Sequential files. The editing commands are primarily line-oriented and operate on an entire file (i.e., no page buffering needs to be taken into account). It is easier to use than EDIT-11. This is the first implementation of the DEC standard editor, which will operate the same way on all DEC operating systems on which it is provided.

2.1Ø RMS-11 Software Package

The RMS-11 (Record Management Services) software package is bundled into the RSTS/E distribution kit. It contains the following components:

Component	Description
RMS11.RTS	A Run-Time System which may be used by programs using RMS software, usually COBOL-11 programs.
RMS11.TSK RMS11.STB	Files associated with the RMS11 Run-Time System. Used by the Task Builder to process the HISEG option.
RMSLIB.OLB	Object Module Library containing RMS software modules.
RMSUTL.OLB	Object Module Library containing RMS utilities modules.
RMSMAC.MLB	Macro Library containing RMS macros used in MACRO-11 programs
RMSBCK.TSK RMSRST.TSK RMSDFN.TSK RMSDEF.TSK RMSDSP.TSK RMSCNV.TSK	RMS-11 utilities. See the RSTS/E RMS-11 Utilities Manual.

2.11 RT-11 Utilities Package

The RT-11 based utilities are compatible with the V3 release of RT-11. Object modules, libraries, and executable programs developed on VØ6C-Ø3 cannot run on previous versions of RSTS/E. Object modules developed on previous versions should run on VØ6C-Ø3 without modification, however, old libraries are not compatible with the new library format. FORTRAN users should consider recompiling their programs to take advantage of the improved FORTRAN compiler.

2.11.1 PIP.SAV

PIP.SAV is the recommended version of the Peripheral Interchange Program. It runs under the RT11 Run-Time System, and provides most of the features of PIPEXT in a user swap maximum of only 8K. It also performs conversion to and from RMS format files; preserves protection code, clustersize, attributes, and Run-Time System name on disk to disk transfers; does the "right thing" on copies to and from DECtape; provides logging of any actions it performs; and allows selection of files based on creation date. Its directory listing is similar to that produced by DIRECT. PIP.SAV does not support DOS disks or the /FORMAT switch for ANSI magtape. Use PIPEXT for these features.

2.11.2 ONLPAT.SAV

ONLPAT.SAV patches Monitor and Run-Time System files. It is based on the INIT PATCH option and contains additional functionality to process command and log files. Section 6.7 of the RSTS/E System Generation Manual describes the use of ONLPAT during system generation; chapter 9 of the RSTS/E System Manager's Guide describes other uses for ONLPAT.

2.12 Documentation

1) The following RSTS/E manuals have been updated:

   RSTS/E System Generation Manual (AA-2669C-TC)
   RSTS/E System User's Guide (DEC-11-ORSUB-A-D)+(DN1)
   RSTS/E System Manager's Guide (DEC-11-ORSMD-B-D)+(DN1)
   RSTS/E Programming Manual (DEC-11-ORPMA-B-D)+(DN1)
   RSTS/E V06C-03 Release Notes (AA-5246B-TC)

2) The following manuals have been added to the RSTS/E documentation kit:

   Introduction to RMS-11 (AA-0001A-TC)
   RSTS/E RMS-11 Utilities User's Guide (AA-0003B-TC)
   RSTS/E User Environment Test Package (AA-5704A-TC)
   RSTS/E Task Builder Reference Manual (AA-5072A-TC)
   DEC Editor Reference Manual (AA-5789A-TC)
   RMS-11 MACRO Programmer's Manual (AA-0002A-TC)
   PDP-11 MACRO-11 Language Reference Manual (AA-5075A-TC)

3) No changes have been made to the other RSTS/E manuals.

2.13 TECO and RNO

TECO and RNO have been available from the DECUS library and are widely used by many users of RSTS/E. They are included on the distribution kit of V06C-03 for the convenience of those users who wish access to these programs.

NOTE

> TECO and RNO are <u>not</u> released products supported by DIGITAL, and the inclusion of these programs on the distribution kit does not imply recommendation or endorsement by DIGITAL regarding their use. There is no commitment by DIGITAL to continue to include these programs in future distributions of RSTS/E.

TECO is a character-oriented text editor which has been implemented on all major DEC operating systems. (However, TECO is not a supported product on any of the PDP-11 operating systems.) TECO has many powerful features as a text editor, and is especially popular among programmers. On the other hand, many users find TECO's set of commands rather cryptic, while others prefer line-oriented editors. For these users, an editor such as the new DEC editor, EDT, is more appropriate.

RNO is a variant of the RUNOFF program (which is part of the RSTS/E library of system programs). RNO is written in MACRO-11, and generally executes much faster than RUNOFF, which is written in BASIC-PLUS. However, RNO is not completely compatible with RUNOFF, and may sometimes produce different results. This version of RNO was originally developed on RSX-11M and was distributed with the V3 release of that system.

2.14 MACRO Programming

User-written MACRO-11 programs are supported in VØ6C-Ø3 under the following constraints:

1) User code may not generate EMT instructions either explicitly or implicitly. Neither RSTS/E Monitor calls, nor the executive calls for the RT-11 or RSX-11 operating systems, may be used. No documentation is provided that describes these calls. They are reserved for use by software supplied by DIGITAL.

2) Any I/O operations to be performed by a user program must use the RMS-11 interface by issuing calls to the appropriate RMS routines. Note that, as FORTRAN does not support the RMS-11 interface, user-written MACRO subroutines may not perform any I/O operations.

   The RMS-11 MACRO Programmer's Manual (AA-5Ø75A-TC) describes how to program using RMS. Although this manual was originally issued for the IAS and RSX-11M operating systems, practically all of the information in the manual is also applicable to RSTS/E. See Release Note article 19.15.1 N for a list of exceptions. This manual will be updated later to include the information for RSTS/E.

2.15 Distribution Media

No new distribution medium has been added in VØ6C-Ø3. DECtape distribution has been discontinued since there has been practically no demand for it.

2.16 Optional Software Supported

Optional software supported by VØ6C-Ø3 includes the following:

COBOL-11	Version 3
BASIC-PLUS-2	Version 1
RSTS/E FORTRAN	Version 2
RPG II	Version 8
SORT-11	Version 2
RMS-11K	Version 1.5
DMS-5ØØ	Version 2
DIBOL/DECFORM	Version 1
APL-11	Version 1
RSTS/E RJ278Ø	Version 2
DATATRIEVE-11	Version 1

The following software is not supported by VØ6C-Ø3:

COBOL-11	Version 2
RPG II	Version 7.Ø6
SORT-11	Version 1
RSTS/E FORTRAN	Version 1 (VØ1C)

Existing systems which run the above software should obtain the current versions. The obsoleted software may be run on VØ6C-Ø3 only by using the requisite Run-Time Systems and utilities provided in VØ6B-Ø2. Note, however, that if the old RT11.RTS is used, the new PIP.SAV will not work, nor will the new MACRO.SAV or LINK.SAV.

2.17 Old Problems Corrected

Various system problems which were found and fixed after the release of RSTS/E Version VØ6B-Ø2 are listed below. The problems are grouped according to the area in which they occurred.

2.17.1 RSTS/E Monitor

All mandatory Software Dispatch Articles up to and including Sequence No. 3.7.14 (i.e., all patches through December, 1977) have been applied to the Monitor.

The problems fixed are:

Message Send/Receive buffer handling code corrects several problems with messages received with an extended data portion.

The block interlock checking code will prevent a job from reading and locking a block that is already locked by another job.

The job swapping algorithm is more equitable when it picks likely candidates for swapping a job out of memory.

The Monitor recovers properly from all types of memory parity errors.

The modem handling code in Terminal Service is changed to correct several problems with modem lines.

Typing ahead on a logged-out terminal no longer bypasses LOGIN's command line checking, eliminating a breach of system security.

A problem with disabling FIP buffering and not flushing all buffered disk blocks is corrected.

Problems with the "remove logical" and "change logical" FIP SYS calls are corrected.

Numerous errors with the RK611/RKØ6 disk device driver are corrected.

2.17.2 BASIC-PLUS Language and Run-Time System

All mandatory Software Dispatch Articles up to and including Sequence No. 4.1.14 (i.e., all patches through December, 1977) have been applied to the BASIC-PLUS Language and Run-Time System.

The problems fixed include:

A COMPILE with an explicit extension overrides the .BAC default extension.

The UNTIL modifer now considers any non-zero value as "TRUE".

The erroneous error message returned with a COMPILE command which specifies an account which doesn't exist has been corrected.

Comment only lines in EXTEND mode are no longer discarded.

The RANDOMIZE routine has been fixed so that the first random number returned can be near 1.

The length of the data message transferred to the receive buffer by the message receive SYS call is correct.

An error produced by the 2-word math package EXP() is fixed.

Several formatting errors with the PRINT-USING statement are corrected.

The 'Exit and Clear Program' SYS call, SYS(CHR$(9%)) no longer produce continuous errors under certain circumstances.

The expression "A% NOT A%" returns the non-fatal error "?Illegal expression".

2.17.3 Commonly Used System Programs (CUSPs)

The Backup Package programs BACCON, BACDEL, BACDIR, BACFRM, BACLST, and BACKTO correctly handle files which must be split across Backup volumes. Corrections involved the way in which file, account, and block count statistics were updated for the STATUS printout and the final listing file. Individually by program:

BACDEL no longer attempts to delete a split file more than once.

BACLST prints out accurate split file sizes.

BACKTO indicates that a transfer has taken place when only part of a split file has been transferred.

BACFRM correctly processes parts of split files or neighboring files under certain circumstances resulting in inappropriate or uninterpretable error messages. It will inhibit an attempt to transfer the latter part of a split file if the transfer of the previous part failed. The procedure by which BACFRM opens files has been changed so that

1) it does not open the latter part of a split file for output if it is contiguous (wiping out the previous part which was transferred)

2) it correctly restores the Run-Time System names for contiguous files

3) it pre-extends a contiguous file which is being superseded in such a way as to take advantage of the space, and

4) it correctly handles an attempt to supersede a protected file.

The following changes have been made to other CUSPs:

BATDEC no longer places the account password in the intermediate command file.

BATRUN looks up the account password for the account the BATCH job is to run under.

COPY can copy all disks.

DSKINT correctly calculates the new bad pack cluster numbers for the new bad block file correctly.

LOGOUT does not enter an infinite loop if a user's disk quota is exceeded.

ODT handles register 1Ø-17 (Octal) offsets correctly.

PIPSML (formerly 8K PIP) no longer aborts when the null device is specified as the output device.

QUMRUN no longer matches all JOBs of a given Jobname regardless of PPN specified during the KILL or MODIFY command to QUE if the user was Privileged.

SHUTUP is more reliable and foolproof.

SPLRUN does a better job of dealing with files that have attributes.

TTYSET no longer includes SPEEDs in most of its MACROs.

2.17.4 Run-Time Systems and Support

RSX.RTS	Several obscure problems in the Run-Time System are fixed. None of these problems were published in the VØ6B Software Dispatch.
TKB.TSK MAC.TSK LBR.TSK	These utilities accept RSTS/E logical device names and special PPN designators (e.g., "$") in filename specifications.

2.18 Upgrading from VØ6B-Ø2 to VØ6C-Ø3

When you are upgrading from RSTS/E VØ6B to RSTS/E VØ6C, you may place your VØ6C SYSGEN system or your VØ6C target system on your VØ6B system disk, or on any other disk which contains a valid RSTS/E file structure. This disk must contain enough contiguous space to accommodate the new system.

Before using a disk as a VØ6C system disk, be sure to create a backup copy. Once you start the initialization code COPY procedure, your disk is no longer usable as a VØ6B system disk.

After backing up the disk, you should create as much contiguous space on it as you can. Using RSTS/E VØ6B timesharing, delete as many files as possible. For example, you may wish to delete all .TMP and .BAK files. Shut down your VØ6B system. Use the REFRESH option of INIT to delete .RTS files in account [Ø,1]. You should delete RT11.RTS and all versions of BASIC-PLUS. You should also delete any run-time systems which will be built when you install optional VØ6C software packages (e.g., RMS11.RTS). Delete all monitor .SIL files (if any).

You should now have as much contiguous space as possible. Bootstrap your VØ6C medium (distribution or intermediate) and use the COPY option to copy system files to the target disk. Specify 'D' to COPY to delete VØ6B system files and replace them with VØ6C files. If this succeeds, you may proceed to tailor your RSTS/E VØ6C system. If there is still not enough contiguous space, you must use a new disk for your VØ6C system. Note that after you have built your VØ6C system, you can use the BACKUP package to rebuild the disk. Be sure to specify no superseding of VØ6C library files.

3.Ø Contents of RSTS/E VØ6C-Ø3 Distribution Kits

For each item, the catalogue number and a description are given. Where applicable, the logical disk name is also given.

Label	Logical Name	Description
		MONITOR
INIT/MONITOR/BASIC-PLUS/RSX/RMS11 (Combined)		
AM-2774F-BC	SYSGNF	RKØ6 DEC Data Cartridge
INIT/MONITOR/BASIC-PLUS		
AP-2772F-BC		TU1Ø/TE1Ø 7-track Magnetic Tape
AP-2773F-BC		TU1Ø/TE1Ø/TSØ3/TU16/TE16/TU45 9-track Magnetic Tape
AN-2771F-BC	SYSGNF	RKØ5 DECpack
		UTILITIES
UTILITY PROGRAMS (Combined)		
AM-5445F-BC	SYSL1F	RKØ6 DEC Data Cartridge
UTILITY PROGRAMS (Standard programs, Spooling programs, Big programs)		
AP-2752F-BC		TU1Ø/TE1Ø 7-track Magnetic Tape
AP-2753F-BC		TU1Ø/TE1Ø/TSØ3/TU16/TE16/TU45 9-track Magnetic Tape
AN-2751F-BC	SYSL1F	RKØ5 DECpack
UTILITY PROGRAMS (BACKUP programs, UETP programs)		
AP-C726F-BC		TU1Ø/TE1Ø 7-track Magnetic Tape
AP-C725F-BC		TU1Ø/TE1Ø/TSØ3/TU16/TE16/TU45 9-track Magnetic Tape
AN-5444F-BC	SYSL2F	RKØ5 DECpack

RUN-TIME SYSTEMS

RSX/RMS Libraries/EDT (Combined)

AN-C885A-BC	RSXLBA	RK05 DECpack

RSX LIBRARY/EDT

AP-C881A-BC		TU10/TE10 7-track Magnetic Tape
AP-C883A-BC		TU10/TE10/TS03/TU16/TE16/TU45 9-track Magnetic Tape

RMS LIBRARY

AP-C882A-BC		TU10/TE10 7-track Magnetic Tape
AP-C884A-BC		TU10/TE10/TS03/TU16/TE16/TU45 9-track Magnetic Tape

Patch Kit

AP-D008A-BC		TU10/TE10 7-track Magnetic Tape
AP-D009A-BC		TU10/TE10/TS03/TU16/TE16/TU45 9-track Magnetic Tape
AN-D010A-BC	PATCHA	RK05 DECpack
AM-D011A-BC	PATCHA	RK06 DEC Data Cartridge

4.Ø Notebook Maintenance

Each month you should take apart the Software Dispatch and file the new articles by sequence number in your notebook. If an article is a replacement, discard the superseded article. Each article addresses a single topic and its title gives the symptom or a one-line problem description. This information appears at the top of each page:

RSTS/E Software Dispatch, (date)

RSTS/E VØ6C-Ø3 Seq 1Ø.21.1 O
System Utilities Package
SHUTUP Page 1 of 2

The System COMPONENTS.SUBCOMPONENTS are listed in Section 4.1. Articles for each subcomponent are assigned a sequence number which uniquely identifies the article within that component category. As each new article is published for a sub-component, it is assigned the next higher sequence number. The sequence numbers are defined as COMPONENT.SUBCOMPONENT.ISSUE number. An example is the 1Ø.21.1 shown in the upper right corner of the sample heading. Replacement articles are noted as such.

Articles for subordinate software that is not unique to RSTS/E are to be filed by product and then component. Articles are sequenced by component after this sorting process. In some cases, there are so few articles that they are sequenced only by product. Missing sequence numbers indicate problems which only apply to other operating systems. The cumulative index is published in the Dispatch to help you maintain the notebook for the complete RSTS/E system.

A flag may appear after the sequence number:

M = Mandatory patch
O = Optional patch (as in the sample heading)
R = Restriction
N = Note

4.1 RSTS/E COMPONENT.SUB-COMPONENT Assignments

The COMPONENT.SUB-COMPONENT numbers outlined below appear on articles to be filed in Section 7.Ø of this notebook.

RSTS/E MONITOR

1.Ø Initialization
- 1.1 INIT.SYS Program Patches
- 1.2 INIT.SYS Program Notes

2.Ø System Generation
- 2.1 SYSGEN Patches
- 2.2 SYSGEN Notes

3.Ø Executive
- 3.1 Monitor Patches
- 3.2 Monitor Notes
- 3.3 Terminal Service Patches
- 3.4 Terminal Service Notes
- 3.5 File Processor Patches
- 3.6 File Processor Notes
- 3.7 Device Driver Patches
- 3.8 Device Driver Notes

4.Ø BASIC-PLUS
- 4.1 BASIC-PLUS Patches
- 4.2 BASIC-PLUS Notes
- 4.3 Programming Hints
- 4.4 Math Packages
- 4.5 Matrix Functions
- 4.6 PRINT-USING
- 4.7 Record I/O
- 4.8 Virtual Core
- 4.9 String Arithmetic

5.Ø RSTS/E 278Ø Device Driver
- 5.1 278Ø Device Driver Patches
- 5.2 278Ø Device Driver Notes

6.Ø (Reserved)

7.Ø (Reserved)

8.Ø (Reserved)

9.Ø (Reserved)

RSTS/E UTILITIES

1Ø.Ø System Utilities Package
- 1Ø.1 Package Notes
- 1Ø.2 BUILD Utility
- 1Ø.3 DIRECT
- 1Ø.4 DISPLY
- 1Ø.5 VTØ5 Adaptation of DISPLY
- 1Ø.6 VT5Ø Adaptation of DISPLY
- 1Ø.7 (Reserved)
- 1Ø.8 DSKINT
- 1Ø.9 GRIPE
- 1Ø.1Ø INIT
- 1Ø.11 INUSE
- 1Ø.12 LOGIN
- 1Ø.13 LOGOUT
- 1Ø.14 MONEY
- 1Ø.15 ODT
- 1Ø.16 PLEASE
- 1Ø.17 PRIOR
- 1Ø.18 QUOLST
- 1Ø.19 REACT
- 1Ø.2Ø REORDR
- 1Ø.21 SHUTUP
- 1Ø.22 SYSCAT
- 1Ø.23 SYSTAT
- 1Ø.24 TALK
- 1Ø.25 TTYSET
- 1Ø.26 UMOUNT
- 1Ø.27 UTILTY
- 1Ø.28 UTILT1
- 1Ø.29 SWITCH

11.Ø Data Manipulation Package
- 11.1 Package Notes
- 11.2 COPY
- 11.3 EDIT
- 11.4 EDITCH
- 11.5 FILCOM
- 11.6 FLINT
- 11.7 PIPSML
- 11.8 PIPEXT
- 11.9 RUNOFF
- 11.1Ø BPCREF
- 11.11 BPCRF1
- 11.12 PMDUMP
- 11.13 FLX
- 11.14 MAKRTS

12.Ø Error Control Package
- 12.1 Package Notes
- 12.2 ANALY1
- 12.3 ANALYS
- 12.4 ERRCPY
- 12.5 ERRDET
- 12.6 ERRDIS
- 12.7 ERRINT
- 12.8 ERRBLD

13.Ø **Backup Package**
- 13.1 Package Notes
- 13.2 BACCOM
- 13.3 BACCON
- 13.4 BACDEL
- 13.5 BACDIR
- 13.6 BACDMP
- 13.7 BACDSK
- 13.8 BACENT
- 13.9 BACFRM
- 13.1Ø BACLAB
- 13.11 BACLOD
- 13.12 BACLST
- 13.13 BACMNT
- 13.14 BACPRM
- 13.15 BACKTO
- 13.16 BACKUP

14.Ø **Spooling and Operator Services Package**
- 14.1 Package Notes
- 14.2 OPSER
- 14.3 OPSRUN
- 14.4 QUE
- 14.5 QUEMAN
- 14.6 QUMRUN
- 14.7 CHARS
- 14.8 SPOOL
- 14.9 SPLIDL
- 14.1Ø SPLRUN
- 14.11 BATDCD
- 14.12 BATCH
- 14.13 BATIDL
- 14.14 BATDEC
- 14.15 BATRUN

15.Ø **User Environment Test Package**
- 15.1 Package Notes
- 15.2 UETP
- 15.3 ACCTST
- 15.4 ACØTST
- 15.5 AC1TST
- 15.6 AC2TST
- 15.7 AC3TST
- 15.8 AC4TST
- 15.9 AC5TST
- 15.1Ø (Reserved)
- 15.11 DBEXER
- 15.12 DPEXER
- 15.13 DREXER
- 15.14 DMEXER
- 15.15 DSEXER
- 15.16 DFEXER
- 15.17 DKEXER
- 15.18 DXEXER
- 15.19 MTEXER
- 15.2Ø DTEXER
- 15.21 CPEXER
- 15.22 CREXER
- 15.23 PPEXER

15.24 PREXER
15.25 KBEXER
15.26 LPEXER
15.27 SCRIPT
15.28 SCRPTB
15.29 CVTSCP
15.3Ø OPTPRM
15.31 OPTPRK
15.32 OPTPRB
15.33 TDLSRT
15.34 TDLXQK
15.35 TDLMQK
15.36 TDLOQK
15.37 TDLSQK
15.38 CRFILE
15.39 TAPSRT
15.4Ø TAPSRU
15.41 DECMRG
15.42 (Reserved)
15.43 VERIFY
15.44 RANMAK
15.45 JSTATS
15.46 DSTATS
15.47 CPUTST
15.48 FILMIN
15.49 CRUDAT
15.5Ø FLOPPY
15.51 CRDATA
15.52 BUILDR
15.53 PKBSNE

16.Ø (Reserved)

17.Ø Automated Patching Facility Package
17.1 Package Notes
17.2 PATCPY
17.3 PBUILD
17.4 CPATCH
17.5 AUTOED
17.6 ONLPAT.SAV

18.Ø (Reserved)

RSTS/E DOCUMENTATION

19.Ø	**Documentation**	
19.1	Documentation Directory	DEC-11-ORDAA-B-D
19.2	RSTS/E System Generation Manual	AA-2669C-TC
19.3	RSTS/E System Manager's Guide	DEC-11-ORSMD-B-D(+DN1)
19.4	RSTS/E System User's Guide	DEC-11-ORSUB-A-D(+DN1)
19.5	BASIC-PLUS Language Manual	DEC-11-ORBPB-B-D
19.6	RSTS/E Programming Manual	DEC-11-ORPMA-B-D(+DN1)
19.7	RSTS/E RUNOFF User's Guide	DEC-11-URUNA-B-D
19.8	RSTS/E Text Editor Manual	DEC-11-UTEMA-A-D
19.9	RSTS/E VØ6C-Ø3 Release Notes	AA-5246B-TC
19.1Ø	RSTS/E Pocket Guide	DEC-11-ORFCA-A-D
19.11	(Reserved)	
19.12	RSTS/E User Environment Test Package	AA-57Ø4A-TC
19.13	RSTS/E Task Builder Reference Manual	AA-5Ø72A-TC
19.14	Introduction to RMS-11	AA-ØØØ1A-TC
19.15	RMS-11 MACRO Programmer's Reference Manual	AA-ØØØ2A-TC
19.16	RSTS/E RMS-11 Utilities User's Guide	AA-ØØØ3B-TC
19.17	DEC Editor Reference Manual	AA-5789A-TC
19.18	PDP-11 MACRO-11 Language Reference Manual	AA-5Ø75A-TC

RUN-TIME SYSTEMS SUPPORT

20.0 (Reserved)

21.0 RSX Emulator Package
- 21.1 Package Notes
- 21.2 (Reserved)
- 21.3 RSX.RTS
- 21.4 TKB.TSK
- 21.5 MAC.TSK
- 21.6 LBR.TSK
- 21.7 SYSLIB.OLB
- 21.8 RSXMAC.SML
- 21.9 PAT.TSK
- 21.10 EDT.TSK

22.0 RT-11 Emulator Package
- 22.1 Package Notes
- 22.2 RT11.RTS
- 22.3 CREF.SAV
- 22.4 EDIT.SAV
- 22.5 HOOK.SAV
- 22.6 LIBR.SAV
- 22.7 LINK.SAV
- 22.8 LOGIN.SAV
- 22.9 LOGOUT.SAV
- 22.10 MACRO.SAV
- 22.11 PAT.SAV
- 22.12 PATCH.SAV
- 22.13 PIP.SAV
- 22.14 SILUS.SAV
- 22.15 SYSBAT.SAV
- 22.16 SYSGEN.SAV
- 22.17 UTILTY.SAV

4.2 Subordinate Software

23.0 FORTRAN IV/RSTS-E V02
- 23.1 Package Notes
- 23.2 COMPILER
- 23.3 OTS
- 23.4 RUNTIME SYSTEM

24.0 DMS-500 V02
- 24.1 Package Notes
- 24.2 DMSUTL
- 24.3 DMSFSS
- 24.4 DMSISR
- 24.5 DMSISC
- 24.6 DMSRAR
- 24.7 DMSISA
- 24.8 DMSRAA
- 24.9 DMSFSU
- 24.10 DMSISO
- 24.11 DMSRAO
- 24.12 DMSFSX
- 24.13 IAMGEN
- 24.14 IAMCRI
- 24.15 IAMDMP
- 24.16 IAMVFY
- 24.17 IAMPRI
- 24.18 IAMFNS
- 24.19 DSORT
- 24.20 DSORTD
- 24.21 DSORTF
- 24.22 DSORTX
- 24.23 DSORTS
- 24.24 DSORTM
- 24.25 DSORTO
- 24.26 DMS-500 User's Guide

25.0 RSTS/E 2780 PACKAGE
- 25.1 Package Notes
- 25.2 RJ2780
- 25.3 RSTS/E 2780 User's Guide (DEC-11-ORJEA-B-D)

26.0 DIBOL
- 26.1 Package Notes

27.0 BASIC-PLUS-2
- 27.1 Package Notes
- 27.2 BASIC2 Compiler Patches
- 27.3 BASIC2 Run-Time System Patches
- 27.4 BP2COM Run-Time System Patches
- 27.5 BASIC2 Object Library Patches
- 27.6 BP2COM Object Library Patches
- 27.7 BASRMS Object Library Patches
- 27.8 TRANSLATOR Utility
- 27.9 BASIC-PLUS-2 Language Manual (AA-0153A-TK)
- 27.10 BASIC-PLUS-2 RSTS/E User's Guide (AA-0154A-TC)
- 27.11 BASIC-PLUS-2 RSTS/E Installation Guide (AA-0156A-TC)

28.0 **COBOL V3**
- 28.1 Package Notes
- 28.2 COBOL Compiler
- 28.3 COBOL Object Library
- 28.4 COBOL ODL
- 28.5 COBOL Utilities
- 28.6 PDP-11 COBOL Language Reference Manual (AA-1749D-TC)
- 28.7 PDP-11 COBOL User's Guide (AA-1757C-TC)
- 28.8 PDP-11 COBOL Installation Guide (AA-1745C-TC)

29.0 **RMS-11**
- 29.1 Package Notes
- 29.2 RMS11.RTS
- 29.3 RMS11.ODL
- 29.4 RMSLIB.MLB
- 29.5 RMSMAC.OLB
- 29.6 RMSUTL.OLB
- 29.7 RSTS/E RMS-11K Installation Guide (AA-5185B-TC)

30.0 **SORT-11 V02**
- 30.1 Package Notes
- 30.2 SORT Utility
- 30.3 PDP-11 SORT Reference Manual (AA-3341C-TC)

31.0 **RSTS/E DECNET/E Package**
- 31.1 Package Notes

32.0 **RPG II V8.00**
- 32.1 Package Notes

33.0 **APL-11**
- 33.1 Package Notes

COBOL-11 V2

SORT-11 V1

RPGII V7.06

COMPILER	V7.06
OTS	V7.06
RPGASN	V1-06
RPGDCV	V1-14
RPGDMP	V1-04
RPGESP	V1-16
RPGFIL	V2-04
RPGFNS	V1-12
RPGPCV	V1-14
RPGXEQ	V1-02

5.0 Software Performance Reports

Each new installation is provided with Software Performance Report (SPR) forms. The SPR form enables the user to suggest enhancements to or report problems with DIGITAL software or documentation. When a problem is encountered, an SPR should be completed and mailed to the local SPR Center (see inside of back cover).

Responses will be sent to the name and address appearing on the form. Additional SPR forms may be obtained by writing to the local SPR Center. SPR response is free for one year after installation and may be continued by subscription thereafter.

5.1 Software Performance Report Guidelines

This guideline for SPR completion is provided to ensure that adequate information is included to prevent delays in processing.

Every SPR form should be completely filled out. A clear description of the problem is very important and will speed response.

It is important to know the complete hardware configuration for the installation, including CPU type, system disk, amount and type of memory, hardware options (such as hardware floating-point processor), and peripherals on the system. Similarly, it is important to know what RSTS/E monitor options were specified at system generation time, including which math package and BASIC-PLUS options were selected. The easiest way to supply this information is to attach a copy of the listing of the CONFIG.MAC and SYSGEN.CTL files produced during the system generation process. Alternatively, a copy of the console terminal dialogue could be provided to show which options and procedures were selected.

In addition to the above points, it is important to specify the following items when submitting an SPR concerning a DIGITAL supplied BASIC-PLUS program:

1. The name of the CUSP.
2. The version and edit level of the CUSP. This is generally found at statement 1010 in the program listing. The optional patches included in the CUSP, if any, should also be listed.
3. Under what account(s) the CUSP failed; distinguish between privileged and non-privileged behavior.
4. The PRIORITY and SWAP MAX which the CUSP was running under when it failed.
5. A terminal printout of the relevant command strings and/or input data.
6. A complete list of any modifications which have been made to the CUSP at the user's installation. (This will not only help to determine the possible cause of failure but may also suggest changes to improve CUSP performance and reliability.)
7. A listing of the contents of log files, where applicable.

Software Support is unable to analyze crash dumps submitted with SPRs unless, as an absolute minimum, listings of the system load maps are included. Submitted crash dumps should include the full "CORE DUMP OF MONITOR" output of the crash dump analysis program. It helps to include, in machine readable form, the [0,1]CRASH.SYS file along with the installed monitor .SIL file (usually [0,1]RSTS.SIL).

Before submitting an SPR, the user should review the relevant section(s) of these Release Notes to ensure that the problem has not already been solved and a solution published.

6.Ø Patching Procedures

This section briefly describes the patching procedures to be used for maintenance of RSTS/E Version VØ6C-Ø3. New to version 6C is an automated patching facility for both the monitor and BASIC-PLUS CUSPs. Programs included in the patching package are ONLPAT for the monitor, and PATCPY, PBUILD, CPATCH, and AUTOED for the CUSPs. The procedures to be followed for patching the monitor and CUSPs when generating a new RSTS/E system are described in section 6.7 of the RSTS/E System Generation Manual. Autopatching is a completely new feature in RSTS/E. The procedures are subject to change in future releases.

Some patches to the monitor or INIT code must be put in manually with the INIT PATCH option before the system may be brought up for using autopatching. Any such patches will be clearly marked in the RSTS/E Software Dispatch or the RSTS/E Release Notes if such special attention is required.

The programs used for automatic patching can also be used for manual patching. The procedures for manual use can be found in the RSTS/E System Manager's Guide. The rest of this section summarizes, briefly, the manual procedures.

Two methods will be used for RSTS/E software: 1) Monitor patches and 2) CUSP patches. Other optional software supplied to RSTS/E installations may require different patching procedures from those about to be discussed; these alternative methods will be explained further under the associated COMPONENT.SUB-COMPONENT section for the software concerned. Unless otherwise stated, all patches published in the Release Notes and the Software Dispatch are MANDATORY patches which must be made to the affected component as soon as possible. Since not all components will be included at each installation, patches to unused components will fail when the attempt is made to apply them. The system manager, however, should file all articles, even for unused components, since the installation might choose at some future time to install additional components and those would then have to be brought up to the latest support level.

Some articles will detail the procedure to patch a component in order for the installation to use an optional feature of that component. These "Optional" patches are not required to keep a component at its current support level; however, if problems occur with a component containing optional features, the installation should be sure to report which optional features are in use when filing the Software Performance Report (SPR).

Next, some discussion of documentation conventions needs to be presented. Below is a table of syntactic elements which provides a description of certain terminal keys which are non-printing functions:

^C or CTRL/C	Control-C
<tab>	Horizontal tab
<lf>	Line feed

<ff>	Form feed
<cr>	Carriage return
^Z or CTRL/Z	Control-Z
<esc>	Escape (Alt Mode or SEL)

When one of these syntactic elements appears in the documentation, it signifies that the associated key(s) is (are) pressed if input is desired or the appropriate terminal action occurs if an output operation is being described.

An optional item this will be indicated by enclosing it between brackets. Note that nesting may take place, as in the following example:

```
RUN <program name>[/DET[ACH]]
```

The modifier /DETACH is optional, as are the last three letters of DETACH. In other words, /DET, /DETA, /DETAC, or /DETACH is acceptable, if this modifier is used.

Note that whenever a project-programmer number is called for in the text, the [and] which may appear are explicitly stated and do not signify that the enclosed quantity is optional.

6.1 Monitor Patching

Changes to the RSTS/E monitor will be made via the PATCH option of INIT.SYS and/or with the new program ONLPAT, as described in the RSTS/E System Generation Manual. Sections 3.4 (PATCH) and 6.7 (ONLPAT) describe in detail the use of this option to make changes to monitor code.

Release Note articles discussing various monitor options and problem corrections will contain the exact operator action required when using the PATCH option. The expected dialogue will be shown in the article as it should appear when the patch is made. If any deviation occurs, the process should be halted, any changes already made returned to their former values via the PATCH option, and a determination made as to where the discrepancy occurred.

6.2 Commonly Used System Program (CUSP) Patching

The RSTS/E CPATCH program will serve as the CUSP program patching tool for maintenance purposes. The program uses a subset of the RSTS/E EDIT program commands to perform patches one line at a time.

Refer to the RSTS/E System Manager's Guide for complete information on the commands and capabilities of CPATCH and PBUILD.

Each patch notice will be self-contained as a convenience to the person who will be making the change. It is recommended that the date of the change and the name of the person making the patch be written on the article at the time the program modification is performed in order to keep an up-to-date record of the state of the system library software.

7.Ø Release Notes and Software Dispatch Articles

Section 7.Ø is the repository for all Release Notes and Software Dispatch articles. This section will include problem solutions, optional feature "patches", documentation errata, programming hints, and tips for better operation of RSTS/E.

Anyone who generates a RSTS/E system (whether on-line under time sharing or as the initial version of the system) should become familiar with these articles. This should be a continuing process of education as sub-sections are added and updated.

Beginning on the next page is an index to all Release Notes Articles. This index will be continually updated as new articles are issued.

7.1 Cumulative Index of all Articles

Component	Sequence	
RSTS/E V06C-03		
RSTS/E INIT.SYS PROGRAM		
INIT.SYS PROGRAM NOTES		
CLEANING DISKS AFTER SYSTEM CRASHES	1.2.1 N	RN
ADDING BAD BLOCKS TO THE BAD BLOCK FILE	1.2.2 N	RN
DEVICES WITH NON-STANDARD UNIBUS ADDRESSES AND VECTORS	1.2.3 N	RN
RSTS/E EXECUTIVE		
MONITOR PATCHES		
MAKING RECEIVER DECLARATION NON-PRIVILEGED	3.1.1 O	RN
FIRST FIT MEMORY ALLOCATION	3.1.2 O	RN
TERMINAL SERVICE PATCHES		
RESTRICTING TERMINAL MODES	3.3.1 O	RN
IMMEDIATE MODEM HANGUP	3.3.2 O	RN
CHANGING SPECIAL LOGIN TERMINAL	3.3.3 O	RN
CHANGING A TERMINAL'S INPUT BUFFER QUOTA	3.3.4 O	RN
INPUT ESCAPE SEQUENCE HANDLING	3.3.5 O	RN
RESTRICTING PSEUDO-KEYBOARD MODES	3.3.6 O	RN
TERMINAL SERVICE NOTES		
2741 HARDWARE OPTIONS REQUIRED BY RSTS/E	3.4.1 N	RN
2741 TERMINAL INTERFACE WIRING	3.4.2 N	RN
UNDERSTANDING SYNCHRONIZATION DELAYS	3.4.3 N	RN
UNDERSTANDING '?I/O TO DETACHED KEYBOARD' ERRORS	3.4.4 N	RN
UNDERSTANDING TERMINAL PARITY AND STOP BIT SETTINGS	3.4.5 N	RN
MODEM CONTROL ON PRIVATE, LEASED LINES	3.4.6 N	RN
FILE PROCESSOR PATCHES		
AUXILIARY LIBRARY ACCOUNTS	3.5.1 O	RN
CHANGING DEFAULT PROTECTION CODE	3.5.2 O	RN
RESTRICTING DISK MODES	3.5.3 O	RN
MAKING FIP SYS CALLS PRIVILEGED	3.5.4 O	RN
NON-PRIVILEGED ACCESS TO NON-FILE STRUCTURED DISKS	3.5.5 O	RN
NON-PRIVILEGED WRITES TO NON-FILE STRUCTURED DISKS	3.5.6 O	RN
ALLOWING CROSS ACCOUNT CREATIONS	3.5.7 O	RN
FILE PROCESSOR NOTES		
WRITE PROTECT AGAINST OWNER FILE PROTECTION CODE HANDLING	3.6.1 N	RN
DEVICE DRIVER PATCHES		
SPECIAL LINE PRINTER DRUMS	3.7.1 O	RN
CHANGING DEFAULT PRINTER FORM LENGTH	3.7.2 O	RN
TU16/TU45/TE45 MAGTAPE DENSITY/PARITY DEFAULT	3.7.3 O	RN
DEVICE DRIVER NOTES		
CR11 CARD READER NOTES	3.8.1 N	RN
RSTS/E BASIC-PLUS		
BASIC-PLUS PATCHES		
SPECIAL PRINT-USING CHARACTERS	4.1.1 O	RN
DEFAULT SCALE FACTOR	4.1.2 O	RN

Component	Sequence	
SCALE FACTOR WARNING MESSAGE	4.1.3 O	RN
DEFAULT TO EXTEND MODE	4.1.4 O	RN
NO IMMEDIATE MODE FROM .BAS FILE	4.1.5 O	RN
NO PPN'S IN CATALOG	4.1.6 O	RN
BASIC-PLUS NOTES		
TIME$ FUNCTION	4.2.1 N	RN
COMMA PRINT ZONE HANDLING	4.2.2 N	RN
CARRIAGE RETURN HANDLING IN INPUT LINE	4.2.3 N	RN
PROGRAMMING HINTS		
HANDLING OF INTEGER AND FLOATING POINT CONSTANTS	4.3.1 N	RN
RSTS/E 278Ø DEVICE DRIVER		
278Ø DEVICE DRIVER NOTES		
INSTALLING AND TROUBLESHOOTING 278Ø'S	5.2.1 N	RN
SYSTEM UTILITIES PACKAGE		
SHUTUP		
CHANGING 'SHUTUP' SLEEP PARAMETERS	1Ø.21.1 O	RN
SHUTUP'S RUN-TIME SYSTEM MUST BE THE SYSTEM DEFAULT	1Ø.21.2 R	RN
DATA MANIPULATION PACKAGE		
PACKAGE NOTES		
HOW TO SAVE FILE ATTRIBUTE DATA AND RUN-TIME SYSTEM NAMES WHEN TRANSFERRING FILES	11.1.1 N	RN
EDITCH		
CHANGING THE EFFECT OF THE POSITION COMMAND	11.4.1 O	RN
ERROR CONTROL PACKAGE		
PACKAGE NOTES		
ERRLOG.FIL FILE USED IN VØ6B SHOULD BE DELETED	12.1.1 N	RN
SPOOLING AND OPERATOR SERVICES PACKAGE		
PACKAGE NOTES		
MANUAL SHUTDOWN OF THE LINE-PRINTER SPOOLING PACKAGE	14.1.1 N	RN
FILES QUEUED WITH WILDCARD SPECIFICATION AND DELETE SWITCH DO NOT GET DELETED	14.1.2 N	RN
WILDCARD FILE LOOKUP OPERATIONS OF THE SPOOLING PACKAGE	14.1.3 R	RN
NOTES ON THE USE OF THE LA18ØS TERMINAL UNDER RSTS/E VØ6C	14.1.4 N	RN
QUMRUN		
DISABLING 'JOB WITH DIFFERENT FORM NAME WAITING' MESSAGE	14.6.1 O	RN
SPLRUN		
ELIMINATING EXTRA FORM FEEDS ON SPECIAL FORMS	14.1Ø.1 O	RN
DOCUMENTATION		
RMS-11 MACRO PROGRAMMER'S REFERENCE MANUAL		
ITEMS NOT AVAILABLE ON 6C IN RMS-11 MACRO MANUAL	19.15.1 N	RN

Component	Sequence	
BASIC-PLUS-2		
BASIC2 COMPILER PATCHES		
VIRTUAL ARRAY ELEMENTS MAY BE MISALIGNED	27.2.1 M	RN
POSSIBLE FATAL ERRORS ON "PRINT-USING" STATEMENTS	27.2.2 M	RN
ERRORS ON ACCESSING VARIABLES WITH DEBUGGER AND "TSK" OUTPUT	27.2.3 M	RN
LARGE TERMINAL WIDTHS PRODUCE FATAL ERRORS	27.2.4 M	RN
POST MORTEM DUMP OPTION ALWAYS ENABLED ON "TSK" OUTPUT	27.2.5 O	RN
BASIC2 RUN-TIME SYSTEM PATCHES		
POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON STRING EQUATES	27.3.1 M	RN
POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON PRINT-USINGS	27.3.2 M	RN
CLOSING A WRITE LOCKED OR HUNG DEVICE RESULTS IN A LOOP	27.3.3 M	RN
CHAINING TO A BASIC-PLUS PROGRAM FROM BASIC-PLUS-2 MAY RESULT IN SCALE FACTOR INTERLOCK	27.3.4 M	RN
ERROR TRAPPING OUT OF STRING FUNCTION GIVES MEM. MGT. VIOLATION	27.3.5 M	RN
LARGE TERMINAL WIDTHS PRODUCE FATAL ERRORS	27.3.6 M	RN
DEBUG MAY IGNORE BREAK IN USER FUNCTION	27.3.7 M	RN
AN ATTEMPT TO USE A SUBSCRIPT OUT OF RANGE IN DEBUG WILL GIVE MEMORY MANAGEMENT ERRORS	27.3.8 M	RN
UNQUOTED STRINGS OR "&" AT END OF DATA STATEMENTS CAUSE READ ERRORS	27.3.9 M	RN
DEVICE "TI:" NOT PROPERLY HANDLED	27.3.10 M	RN
ILLEGAL FILENAME MAY NOT GIVE ERROR	27.3.11 M	RN
POSSIBLE BAD DATA IF VIRTUAL ARRAY READ CAUSES ERROR	27.3.12 O	RN
PRINT USING IGNORES TRAILING ASTERISK	27.3.13 M	RN
BP2COM RUN-TIME SYSTEM PATCHES		
USE OF BP2COM RTS MAY GIVE TASK BUILDER ERRORS	27.4.1 M	RN
POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON STRING EQUATES	27.4.2 M	RN
ERROR TRAPPING OUT OF STRING FUNCTION GIVES MEM. MGT. VIOLATION	27.4.3 M	RN
DEVICE "TI:" NOT PROPERLY HANDLED	27.4.4 M	RN
CHANGING PROMPT FOR BP2COM WHEN DEFAULT RUN-TIME SYSTEM	27.4.5 O	RN
CHAINING TO A BASIC-PLUS PROGRAM FROM BASIC-PLUS-2 MAY RESULT IN SCALE FACTOR INTERLOCK	27.4.6 M	RN
BASIC2 OBJECT LIBRARY PATCHES		
DEBUG MAY IGNORE BREAK IN USER FUNCTION	27.5.1 M	RN
AN ATTEMPT TO USE A SUBSCRIPT OUT OF RANGE IN DEBUG WILL GIVE MEMORY MANAGEMENT ERRORS	27.5.2 M	RN
BP2COM OBJECT LIBRARY PATCHES		
POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON STRING EQUATES	27.6.1 M	RN
POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON PRINT-USINGS	27.6.2 M	RN
CLOSING A WRITE LOCKED OR HUNG DEVICE RESULTS IN A LOOP	27.6.3 M	RN
ERROR TRAPPING OUT OF STRING FUNCTION GIVES MEM. MGT. VIOLATION	27.6.4 M	RN
LARGE TERMINAL WIDTHS PRODUCE FATAL ERRORS	27.6.5 M	RN
DEBUG MAY IGNORE BREAK IN USER FUNCTION	27.6.6 M	RN
AN ATTEMPT TO USE A SUBSCRIPT OUT OF RANGE IN DEBUG WILL GIVE MEMORY MANAGEMENT ERRORS	27.6.7 M	RN
BASRMS OBJECT LIBRARY PATCHES		
SCRATCH STATEMENT PRODUCES UNIBUS TIMEOUT ERRORS	27.7.1 M	RN
FILE ACTIVITY AFTER AN INDEXED OPEN MAY PRODUCE CERTAIN RMS ERRORS	27.7.2 M	RN
RPG-II V7.06		
RPG II V7.06 CAN NOT BE INSTALLED ON RSTS/E V06C-03	1 N	RN

CLEANING DISKS AFTER SYSTEM CRASHES

In RSTS/E VØ6C-Ø3, system crash processing operates as follows:

1. Volatile memory is written to [Ø,1]CRASH.SYS after every crash if the crash dump facility was enabled at system startup, regardless of the setting of the console switches.

2. The system performs an automatic restart if the console switches were set to 177777, if the switch register was disabled with the HARDWARE option of INIT, or if the switch register was volatile and the crash was due to a power failure.

If the conditions for performing an automatic restart are not met, or if the system crashed twice within one minute, the CPU halts at address 54. If auto-restart is enabled, the initialization program INIT.SYS is booted, the system disk cleaned, and the INIT.BAC program entered to perform the commands in the $CRASH.CTL file.

On a multi-disk system, commands in $CRASH.CTL will attempt to mount the remaining disks of the public structure. INIT.BAC automatically executes the CLEAN SYS function call as needed.

There is one disadvantage to using the auto-restart facility after system crashes: the code executed by the (on-line) CLEAN SYS function call must be relatively small, and thus its consistency checks are not as extensive as those in the CLEAN function of the INIT.SYS REFRESH option. Running stand-alone, the latter performs extensive validation of the disk file structure and allows you to fix corrupted structures.

DIGITAL recommends that systems run with auto-restart enabled to provide fast, unattended crash recovery. Whenever you shut down the system (such as for preventive maintenance), you should use the CLEAN suboption of REFRESH to verify the file structure on all disks. Such checking ensures that no file system inconsistencies exist. This is especially important if there has been a recent system crash and auto-restart.

ADDING BAD BLOCKS TO THE BAD BLOCK FILE

The ERRDIS program prints a list of potential bad blocks. If you decide to add a block to the bad block file on a disk, you may use the BADS suboption of the REFRESH option during initialization as described in the System Generation Manual, section 3.7.6.

DIGITAL recommends that you follow the procedure below:

1. Use PIP to copy the data in the bad file to a new file. Specify the /GO switch (along with other necessary switches, such as /CL, etc.) to ignore possible USER DATA ERROR ON DEVICE errors. Rename the file which contains the bad block, but do not delete it.:

```
RUN $PIP
*FILE.NEW=FILE.OLD/GO
*FILE.BAD=FILE.OLD/RE
*FILE.OLD=FILE.NEW/RE
```

2. When convenient, shut down the system and use the BADS suboption of REFRESH to add any bad blocks to the disk's bad block file. Each new bad block is now allocated to two files ([Ø,1]BADB.SYS and FILE.BAD in the above example).

3. The CLEAN suboption automatically rebuilds the allocation tables. CLEAN will indicate that the file has a bad block and allow you to delete the file (FILE.BAD in the above example). This is described in section 3.7.2 of the System Generation Manual.

When CLEAN deletes the file, it frees the non-bad blocks in the file.

The above procedure must be modified slightly if the bad block is in a file directory. In this case, use the BACKUP program to copy all accessable files in the account (or disk) to an archive medium. When you add the bad block and clean the disk, the CLEAN suboption will allow you to delete the account whose directory contains the bad block. After starting time sharing, you should use REACT to enter the account on the disk and use BACKUP to restore the files from the archive medium.

DEVICES WITH NON-STANDARD UNIBUS ADDRESSES AND VECTORS

Some installations have devices which are installed at non-standard UNIBUS addresses or which vector to non-standard locations. INIT does not automatically recognize these devices. Of course, the best solution is to re-configure these devices so that they conform to the standard. Appendix F of the System Generation Manual (AA-2669C-TC) contains the information you need to determine the correct UNIBUS addresses and vectors for all RSTS/E devices.

If re-configuration is not possible, you can use the HARDWR option of INIT. The CSR suboption sets non-standard UNIBUS addresses and the VECTOR suboption sets non-standard vectors. DIGITAL strongly recommends that you use only the CSR suboption. This tells INIT to find the device at its non-standard UNIBUS address, but lets INIT determine the device's vector. If the device is functioning properly, INIT will find the device's non-standard vector. The only device for which INIT cannot determine a vector is the card reader. If a card reader has a non-standard vector, you must use the VECTOR suboption.

Be careful when you assign non-standard vectors to devices. INIT checks all device vectors (assigned through the VECTOR suboption or automatically determined) against a table of reserved locations. INIT also checks to see if any vector location is used by more than one device. Any conflict will print the message

Vector for Device XXn: (vvv) already in use - device disabled.

where vvv is the octal address of the erring vector.

The RSTS/E reserved locations, which may not be used as device vectors, are:

Addresses	RSTS/E usage
0-2	Detection of jumps to 0 and traps to 0
4-36	System trap vectors
40-56	Reload start addresses, failure HALT
100-102	KW11-L line frequency clock vector
104-106	KW11-P crystal clock vector
110-112	Jump to 0 handling
114-116	Memory parity trap vector
144-146	Crash dump handling
234-236	Statistics handling
240-242	PIRQ trap vector
244-246	FPP or FIS exception trap vector
250-252	Memory Management Unit trap vector

RSTS/E VØ6C-Ø3
RSTS/E Executive
Monitor Patches

MAKING RECEIVER DECLARATION NON-PRIVILEGED - OPTIONAL MONITOR PATCH

The RSTS/E monitor normally requires that a job have permanent or temporary privileges in order to declare itself to be a message receiver. There are some production environments in which it is useful for non-privileged jobs to have this capability. The following procedure allows you to patch the RSTS/E monitor to remove the privileged status of the message receiver declaration SYS call.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Bootstrap your system disk and use the PATCH option as shown below. If you made the send/receive code resident when you generated the monitor SIL, answer the 'Module name?' question with "FIP". Otherwise, answer with "OVR" as shown.

```
Option: PATCH
File to patch? <lf>      (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø1
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!!
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ^Z                        (CTRL/Z for new module)
Module name? OVR         (or FIP)
Base address? SNDØST
Offset address? ..RCVP-SNDØST
 Base   Offset  Old       New?
??????  ØØØ736  ØØ1445  ? 24Ø
??????  ØØØ74Ø  Ø1Ø446  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E Executive
Monitor Patches

FIRST FIT MEMORY ALLOCATION - OPTIONAL MONITOR PATCH

When a residency request is made for a job or for a Run-Time System residency without a specific load address, the RSTS/E monitor does a best fit allocation: the job or Run-Time System is loaded into the smallest free area in memory in which it fits. For some systems, especially those with a large user memory region, a first fit allocation may result in better performance and less memory fragmentation.

This optional monitor patch changes the memory allocation from best fit to first fit.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Bootstrap your system disk and use the PATCH option as shown below.

```
Option: PATCH
File to patch? <lf>     (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø1
Offset address? Ø
 Base    Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!2
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ^Z                        (CTRL/Z for new module)
Module name? EMT
Base address? ..FFIT
Offset address? Ø
 Base    Offset  Old       New?
133272  ØØØØØØ  ØØ1647  ? 647
133272  ØØØØØ2  Ø2Ø2Ø3  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3 Seq 3.3.1 0
RSTS/E Executive
Terminal Service Patches Page 1 of 1

RESTRICTING TERMINAL MODES - OPTIONAL TERMINAL SERVICE PATCH

The RSTS/E monitor normally allows any user to OPEN a terminal in binary mode or echo control mode (if configured). There is a danger in the use of these terminal modes, since all characters (including CTRL/C) are passed to the user program. A novice user could easily find himself in a situation in which his terminal does not respond. A malicious user could "lock up" a terminal in a way which required the system manager to kill his job. The following procedure allows you to patch the RSTS/E monitor to specify which terminal OPEN modes require privileges. Privileged users and non-privileged users running privileged programs would still have full access to all available terminal modes. Attempts by non-privileged users to use restricted modes would be rejected with no error message.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Decide which terminal modes you wish to make privileged. Consult the RSTS/E Programming Manual for the various modes which are available, and determine the associated MODE values. Compute the sum, n, of the MODE values corresponding to the modes you wish to restrict. If you calculate the sum of the mode values in decimal, include a decimal point after the value of n in the patch below so that it is interpreted as a decimal number.

3. Bootstrap your system disk and use the PATCH option as shown below.

```
Option: PATCH
File to patch? <lf>     (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø3
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!1
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ^Z                        (CTRL/Z for new module)
Module name? TER
Base address? ..KBMP
Offset address? -2
 Base   Offset  Old       New?
??????  177776  Ø42ØØ2  ? <lf>          (no change; verify only)
??????  ØØØØØØ  ØØØØØØ  ? n             (from step 2)
??????  ØØØØØ2  ??????  ? ^C            (CTRL/C to exit)

Option:
```

IMMEDIATE MODEM HANGUP - OPTIONAL TERMINAL SERVICE PATCH

When the carrier frequency drops on a modem line connecting a terminal to RSTS/E, the monitor normally waits approximately 5 seconds before hanging up the line. This allows recovery from transient line failures or accidental unseating of a phone from an acoustic coupler. In certain areas, especially in the United Kingdom and Europe, computer systems are required to disconnect from such a line within milliseconds. The following patch causes RSTS/E to hang up a modem line as soon as it detects loss of carrier signal on that line.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL that supports modem control.

2. Bootstrap your system disk and use the PATCH option as shown below.

```
Option: PATCH
File to patch? <lf>      (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø3
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!2
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ^Z                        (CTRL/Z for new module)
Module name? TER
Base address? ..UKMD
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ØØØ4Ø4  ? 24Ø
??????  ØØØØØ2  Ø127Ø4  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E Executive
Terminal Service Patches

Seq 3.3.3 0

CHANGING SPECIAL LOGIN TERMINAL - OPTIONAL TERMINAL SERVICE PATCH

The number of jobs that can log in to a RSTS/E system is limited by the swapping space available, the JOB MAX set at system start-up time, and the login setting (set by the LOGINS, NO LOGINS, and SET LOGINS functions of UTILTY). However, the console terminal (KBØ:) is a special terminal, and can log in regardless of the login setting, provided that swapping space and JOB MAX permit.

You can change the number of the special keyboard or disable this feature entirely by installing the patch given below.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Decide which terminal should be allowed to log in regardless of the login setting, and use that number as n in the patch below. If you wish to disable this feature, use -1. Be sure you include the decimal point after the new value of n so that it is interpreted as a decimal number.

3. Bootstrap your system disk and use the PATCH option as shown below.

```
Option: PATCH
File to patch? <lf>     (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø3
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!4
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ^Z                        (CTRL/Z for new module)
Module name? TER
Base address? ..CTY.
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ØØØØØØ  ? n.*2 (or -1)  (from step 2)
??????  ØØØØØ2  ØØ14Ø4  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3 Seq 3.3.4 0
RSTS/E Executive
Terminal Service Patches Page 1 of 1

CHANGING A TERMINAL'S INPUT BUFFER QUOTA - OPTIONAL TERMINAL SERVICE PATCH

Inside every terminal's DDB (Device Data Block) is an "input buffer quota" value. The default value is 6. Since there are 3Ø characters in a buffer, this is equal to 6 times 3Ø or 18Ø characters. The terminal service will attempt to buffer that many characters before telling the terminal to stop (by sending it an XOFF). You can increase the input buffer quota value for one or more high speed input terminals by using this optional patch.

NOTE

There is no guarantee that a terminal will be able to get its full buffer quota. A heavy system load may leave less than the terminal's full buffer quota available.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Decide which terminal will have its buffer quota altered, and use that number as n in the patch below. Be sure you include the decimal point after the value of n so that it is interpreted as a decimal number. Also decide what the new buffer quota for that terminal should be, and use that number as m in the patch below.. The legal values for m are 6. through about 2Ø. inclusive.

3. Bootstrap your system disk and use the PATCH option as shown below.

```
Option: PATCH
File to patch? <lf>      (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø3
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!1Ø
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? KBDDDB
Offset address? DDS.KB*n.+26            (n is the KB number)
 Base   Offset  Old       New?
??????  ??????  ØØØØØ6  ? m.            (from step 2)
??????  ??????  ØØØØØØ  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E Executive
Terminal Service Patches

INPUT ESCAPE SEQUENCE HANDLING - OPTIONAL TERMINAL SERVICE PATCH

RSTS/E VØ6C-Ø3 allows multiple character incoming escape sequences. Section 4.4.2 of the RSTS/E Programming Manual (DEC-11-ORPMA-B-D+DN1) fully and correctly describes the escape sequence handling.

One of the multi-character escape sequences is ESCape followed by "P" followed by one modifier character. On the VT52 terminal, the upper left hand key on the auxiliary keypad (usually blue) generates ESC P. If the terminal user types this key and then any other graphic key, the whole sequence is gathered by the terminal service and delivered to the application program. None of the sequence is echoed. Thus the blue key can be considered a "function" key to be followed by some function code character.

On the other hand, some existing applications using the VT52 may expect this key to generate a standalone escape sequence with no required (or desired) modifier character. You may install the following patch to change the terminal service so that the ESCape, "P" sequence does not expect a modifier character. Note that this patch affects <u>all</u> terminals on the system.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Bootstrap your system disk and use the PATCH option as shown below.

```
Option: PATCH
File to patch? <lf>      (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø3
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!2Ø
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ^Z                        (CTRL/Z for new module)
Module name? TER
Base address? ..ESCP
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  Ø5ØØØ1  ? Q&177400
??????  ØØØØØ2  ??????  ? ^C            (CTRL/C to exit)

Option:
```

RESTRICTING PSEUDO-KEYBOARD MODES - OPTIONAL TERMINAL SERVICE PATCH

The RSTS/E monitor normally allows any user to OPEN a pseudo-keyboard with mode 1%, which causes the controlled job to detach rather than be killed if the controlling job closes the pseudo-keyboard prematurely. There is a danger in the use of this mode, since a non-privileged user could fill the system with detached jobs. The following procedure allows you to patch the RSTS/E monitor to specify which pseudo-keyboard OPEN modes require privileges. Privileged users and non-privileged users running privileged programs would still have full access to all available pseudo-keyboard modes. Attempts by non-privileged users to use restricted modes would be rejected with no error message.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Decide which pseudo-keyboard modes you wish to make privileged. Consult the RSTS/E Programming Manual for the various modes which are available, and determine the associated MODE values. Compute the sum, n, of the MODE values corresponding to the modes you wish to restrict. If you calculate the sum of the mode values in decimal, include a decimal point after the value of n in the patch below so that it is interpreted as a decimal number.

3. Bootstrap your system disk and use the PATCH option as shown below.

```
Option: PATCH
File to patch? <lf>     (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø3
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!4Ø
??????  ØØØØØ2  ??????  ? ^Z           (CTRL/Z for new offset)
Offset address? ^Z                     (CTRL/Z for new base)
Base address? ^Z                       (CTRL/Z for new module)
Module name? TER
Base address? ..PKMP
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ØØØØØØ  ? n            (from step 2)
??????  ØØØØØ2  ??????  ? ^C           (CTRL/C to exit)

Option:
```

2741 HARDWARE OPTIONS REQUIRED BY RSTS/E

RSTS/E supports 2741-compatible terminals connected through DL11D, DL11E, and DC11 single-line interfaces and through DH11 and DZ11 multiplexors. Certain 2741 hardware options are required for proper operation under RSTS/E.

The ability to detect "reverse break" is an option on 2741 terminals supplied by some manufacturers, including IBM. A reverse break is a control sequence sent by the computer to a terminal which is currently transmitting (keyboard is unlocked). The purpose of the reverse break is to force the terminal into receive state (keyboard locked). RSTS/E sends a reverse break any time the system has output for the 2741 terminal but internal status tables indicate that the terminal is in transmit state. If the terminal does not recognize the reverse break sequence, lock the keyboard, and switch to receive state, the output will be lost and the terminal can end up in a strange state.

IBM refers to the reverse break detection capability on the IBM 2741 Model 1 Communications Terminal as Feature #4708 - Receive Interrupt. This option must be installed on all IBM 2741 terminals for correct operation under RSTS/E.

The ATTENTION key is also optional on IBM 2741 terminals and is required for operation under RSTS/E. The ATTN key generates a break which is interpreted in several ways by RSTS/E software. The RSTS/E System User's Guide describes the various functions of the ATTN key. IBM refers to the ATTN key and the associated break generation hardware as Feature #7900 - Transmit Interrupt. This option is commonly included on 2741 terminals supplied by IBM and is a standard feature on many 2741-compatible terminals supplied by other manufacturers.

2741 TERMINAL INTERFACE WIRING

The DL11D and DL11E terminal line interfaces require a special configuration for supporting the 2741. The configuration requires 6 data bits, 1 start bit, 1 stop bit, odd parity, 134.5 baud, and rotary switch position 3 for both input and output (full counter-clockwise is position 1). The following jumpers apply to each condition:

Condition	Jumper	
6 Data bits	NB1	out
	NB2	in
1 Stop bit	2SB	in
	J9	out
	J1Ø	in
	J11	out
Odd parity	NP	in
	EPS	in

The 134.5 baud specification requires a 1.Ø3296M crystal (DEC part #18-Ø55Ø2-6). Refer to the DL11 Installation Procedure for specifications pertinent to the DL11D and DL11E.

UNDERSTANDING SYNCHRONIZATION DELAYS

DIGITAL's video terminals (VT5Ø, VT52, etc.) use two special control characters (commonly called XOFF and XON) to request the host computer to suspend and resume data transmission. One use of this synchronization protocol allows the terminal to "freeze" a screen of data giving the user a chance to read it before it has scrolled off the screen. This special mode, called HOLD SCREEN MODE, is enabled and disabled with escape sequences. Once enabled, the terminal detects when a line is about to be scrolled from the screen. The character received from the host computer that would cause the scroll is line feed (LF). At this point, the terminal sends an XOFF to the host and waits for the user to type the SCROLL key. During this time, the terminal will accept and buffer in a temporary storage area (called a silo) further characters received from the host. Even if the host computer can cease transmission with no software delays, there are delays in line transmission that cause a determinable number of characters to be received by the terminal after it has sent the XOFF.

HOLD SCREEN MODE is just one use of this synchronization protocol. Another use is in the handling of the hard copy option of the VT5Ø series terminals. The copier is slow (at least compared to the video screen) and the terminal must ask the host computer to suspend transmission so that it doesn't miss data when the copier is running.

When the LA18Ø DECwriter I is connected to the computer through a serial line interface (referred to as the LA18ØS), it also uses this synchronization protocol. If the LA18ØS "gets behind" due to high line transmission speed or carriage restore time, it requests the host to suspend data transmission. The printer will also send an XOFF if its ONLINE/OFFLINE switch is placed into the OFFLINE position or if it runs out of paper.

The RSTS/E terminal service processing of the synchronization protocol is enabled and disabled by the TTYSET commands STALL and NO STALL.

Below is the formula used for calculating the largest number of characters that would be received after the terminal has sent the XOFF:

$$1 + (3*Y/X) + 2*D*Y$$

> Note: This formula assumes that there is no host computer software delay. In reality there is always some delay at the host computer. This delay might be very small or possibly large depending on system loading and other I/O interrupt activity.

In this formula, X is the terminal's transmission rate in characters per second (CPS), Y is the host computer's transmission rate in CPS, and D is the delay time in seconds for a character to travel from one end of the line to the other.

Considering the formula from left to right:

The 1 is the character storage requirement for the line feed that triggered the XOFF in the first place.

The 3*Y/X is due to the nature of the UART (Universal Asynchronous Receiver/Transmitter) in the terminal. Since the UART is normally double buffered, there can be up to two characters already in the UART's buffers when the terminal decides to send the XOFF. Those two characters plus the XOFF correspond to three character times of terminal to host transmit during which the host may be sending to the terminal. Therefore, it is the ratio of the host and terminal transmission rates that is important for three terminal transmission times.

The 2*D is the total time delay for the line in both directions (the 2) to clear. During this time the host computer may have sent up to 2*D*Y characters to the terminal.

The formula presented here is not completely correct for the VT5Ø series of video terminals. The VT5Ø does not double buffer transmission to the host computer in its UART. On the other hand, the formula can be used as a worst case formula to ensure correct operation of the XON/XOFF Synchronization Protocol.

The following tables give the required size of the temporary storage area (silo) given: 1) the transmission speed (baud rate) from the host computer to the terminal, 2) the transmission speed from the terminal to the host computer, and 3) the physical delay of the transmission line. Local terminal connections have essentially a zero delay factor. The Telephone Company specifies a nominal worst case delay of 5Øms in a coast-to-coast connection, assuming it is not via satellite.

The horizontal scale of baud rates is the baud rate from the host computer to the terminal.

The vertical scale of baud rates is the baud rate from the terminal to the host computer. Baud rates can be converted to characters per second (CPS) by dividing the baud rate by 1Ø (1 start bit + 8 data bits + 1 stop bit = 1Ø bits) with the exception of 11Ø baud which corresponds to 1Ø CPS (1 start bit + 8 data bits + 2 stop bits).

Øms Line Delay

	11Ø	15Ø	3ØØ	6ØØ	12ØØ	24ØØ	48ØØ	96ØØ
11Ø	4	6	1Ø	19	37	73	145	289
15Ø	3	4	7	13	25	49	97	193
3ØØ	2	3	4	7	13	25	49	97
6ØØ	2	2	3	4	7	13	25	49
12ØØ	1	1	2	3	4	7	13	25
24ØØ	1	1	1	2	3	4	7	13
48ØØ	1	1	1	1	2	3	4	7
96ØØ	1	1	1	1	1	2	3	4

50ms Line Delay

	110	150	300	600	1200	2400	4800	9600
110	5	7	13	25	49	97	193	385
150	4	6	10	19	37	73	145	289
300	3	4	7	13	25	49	97	193
600	3	3	6	10	19	37	73	145
1200	2	3	5	9	16	31	61	121
2400	2	3	4	8	15	28	55	109
4800	2	3	4	7	14	27	52	103
9600	2	3	4	7	13	26	51	100

By referring to the manual that came with your terminal you can determine which baud rate combinations are legal for your terminal.

RSTS/E VØ6C-Ø3
RSTS/E Executive
Terminal Service Notes

UNDERSTANDING '?I/O TO DETACHED KEYBOARD' ERRORS

The error '?I/O to detached keyboard' (ERR = 27) occurs only when:

1. one of the simple terminal SYS calls (function codes Ø, 1, 2, 3, 4, and 11) is attempted for the job's console terminal (i.e., the job's KB: device) and the job is detached, or

2. an open is attempted using the device name "KB:" and the job is detached, or

3. an I/O operation (INPUT, PRINT, GET, PUT, etc.) is attempted to a modem line terminal which is neither the job's console terminal nor the terminal from which the job detached and that line is hung up. (Note: Opening a modem line which is currently hung up does not produce any error.)

If an I/O request is issued to a detached job's KB: terminal or any channel on which that terminal is open, the job is put into the hibernate (HB) state. This can happen using local terminals if the job detaches. It can happen on modem lines if the job detaches or the line is hung up and the terminal service automatically detaches the job.

UNDERSTANDING TERMINAL PARITY AND STOP BIT SETTINGS

Terminal Parity

The TTYSET commands "NO PARITY", "EVEN PARITY", and "ODD PARITY" control the format in which data characters are sent to a terminal. NO PARITY (the default) sends the full 8 bits of a byte out to a terminal. EVEN PARITY and ODD PARITY trim the byte to 7 bits of data and then set or clear the eighth bit to achieve the correct parity setting. (Note: Binary Output Mode is "Data Binary Output Mode". NO PARITY yields 8 bit data, EVEN PARITY and ODD PARITY yield 7 bit data.)

The KL11, LC11, DL11A/B/C/D/E, DC11, and DJ11 interfaces do not have programmable hardware parity generation. For these interfaces, the terminal service software generates the output parity bit. The DH11 and DZ11 have programmable hardware parity generation, which the terminal service uses.

Stop Bit Settings

Terminal interfaces with programmable baud rate settings (DC11, DH11, and DZ11) also have programmable stop bit settings. RSTS/E will automatically select 2 stop bits for speed settings of 11Ø baud or less and 1 stop bit for speeds greater than 11Ø baud. Older mechanical terminals such as the ASR33 (which runs at 11Ø baud) require the time delay of 2 stop bits for synchronization. Modern terminals need only 1 stop bit.

When converting from baud rates to characters per second (CPS), the correct number of stop bits becomes clear. Asynchronous communication consists of 1 start bit, 8 data bits, and 1 or 2 stop bit(s).

(11Ø bits per second)/(1+8+2 bits per character)=11Ø/11=1Ø characters per second

(3ØØ bits per second)/(1+8+1 bits per character)=3ØØ/1Ø=3Ø characters per second

MODEM CONTROL ON PRIVATE, LEASED LINES

Some RSTS/E installations use private, leased phone lines for terminal connections. If the line has no modem or modem control (i.e., data signals only) then the line should be set as a local line using the INIT SET option. In this case the private, leased line is simply a long local connection. On the other hand, many private, leased lines do use modem control.

When using modem control, there are five (5) modem signals used by RSTS/E:

1) Ring Indicator (Circuit CE, pin 22)

   This signal is true if the modem's "phone" is ringing. RSTS/E does not control this signal; the modem does. Private, leased lines normally do not "ring".

2) Carrier Detect (Circuit CF, pin 8)

   This signal is true when the modem is receiving carrier from the modem on the other end of the line. RSTS/E does not control this signal; the modem does.

3) Clear to Send (Circuit CB, pin 5)

   This signal is true when it is "O.K. to send data" to the other modem. RSTS/E does not control this signal; the modem does.

4) Data Terminal Ready (Circuit CD, pin 2Ø)

   This signal is controlled by RSTS/E, not the modem. When RSTS/E sets this signal to true, the modem answers the phone if it is not already answered and sends carrier to the modem at the other end of the connection.

5) Request to Send (Circuit CA, pin 4)

   This signal is controlled by RSTS/E, not the modem. When RSTS/E sets this signal to true, the modem tells the other modem it is now "Clear to Send".

RSTS/E always sets and clears Data Terminal Ready and Request to Send together.

The normal sequence for a dial-up modem line is:

1) Someone calls in. The modem asserts Ring Indicator.

2) RSTS/E sets Data Terminal Ready and Request to Send.

3) The modem answers the phone and sends out carrier and Clear to Send. The Ring Indicator goes off.

4) The other end sends carrier and Request to Send. The modem sets Carrier Detect and Clear to Send.

5) RSTS/E enables the line for data communication.

6) Data communication occurs...

7) The other end drops carrier. The modem clears Carrier Detect and Clear to Send.

8) RSTS/E starts a 5 second timeout.

9) The timeout expires. RSTS/E clears Data Terminal Ready and Request to Send.

1Ø) The connection is now broken.

In addition, RSTS/E polls all modem lines once each second. If Carrier Detect has come on without a Ring Indication, the line will be enabled by setting Data Terminal Ready and Request to Send. If Carrier Detect is on when the system is initially started, this happens immediately. So, to use a modem controlled private, leased line as a local line:

1) Strap the Ring Indicator signal false. This ensures that RSTS/E never sees a "ringing" line to get confused about.

2) Strap the Carrier Detect and Clear to Send signals true. This ensures that RSTS/E will immediately enable the line by setting Data Terminal Ready and Request to Send.

RSTS/E VØ6C-Ø3
RSTS/E Executive
File Processor Patches

Seq 3.5.1 0

AUXILIARY LIBRARY ACCOUNTS - OPTIONAL FILE PROCESSOR PATCH

The standard system library for RSTS/E systems is account [1,2]. This account is normally referenced by the $ character in file specifications. The special characters ! % and & can also be used to reference three auxiliary libraries. The default assignments for these characters are [1,3], [1,4], and [1,5] respectively. Although the association of character with account is defined, the three accounts must be created with REACT before the special characters can be used successfully.

In some installations, it may be advantageous to change the account numbers referenced by the three special characters. The account numbers are defined by four words in the RSTS/E monitor. The default account numbers are altered with a simple patch as described below.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Decide which account numbers you wish to use for the auxiliary library characters. You may change any or all of the default assignments (although changing the assignment of the $ library character is strongly discouraged). To specify a new project-programmer number of [p,pn], type a new value at the appropriate offset. The new value has the form:

   p.*256.+pn.

   where p is the new project number (followed by a decimal point to ensure that it is interpreted as a decimal number), the multiplication shifts the project to the high byte (again, do not omit the decimal point after the 256), and pn is the new programmer number (followed by a decimal point). p and pn must be between Ø and 254, and p may not be zero.

3. Bootstrap your system disk and use the PATCH option as shown below. In this example, the ! account is patched to [1ØØ,25Ø], the % account is left unchanged, and the & account is changed to [1,99]. Of course, you should type in appropriate values for your system.

```
Option: PATCH
File to patch? <lf>     (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$0305
Offset address? 0
 Base    Offset  Old       New?
??????  000000  ??????  ? Q!1
??????  000002  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                     (CTRL/Z for new base)
Base address? ..PPN.
Offset address? 0
 Base    Offset  Old       New?
??????  000000  000402  ? <lf>                  (No change to $)
??????  000002  000403  ? 100.*256.+250.       (New value for !)
??????  000004  000404  ? <lf>                  (No change to %)
??????  000006  000405  ? 1.*256.+99.          (New value for &)
??????  000010  ??????  ? ^C                   (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E Executive
File Processor Patches

Seq 3.5.2 0

CHANGING DEFAULT PROTECTION CODE - OPTIONAL FILE PROCESSOR PATCH

When a user creates a file without specifying a protection code for that file, and if the personal default protection code is not set, the RSTS/E monitor creates that file with the system default protection code. This is normally 6Ø., which allows the user read/write access to the file, but restricts all non-privileged users from any access. The legal value range for the default protection code is Ø. to 63. inclusive. The following procedure allows you to patch the RSTS/E monitor to change your system's default protection code.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Bootstrap your system disk and use the PATCH option as shown below. If you made the disk utility code resident when you generated the monitor SIL, answer the 'Module name?' question with "FIP"; otherwise, answer with "OVR" as shown. Be sure you include the decimal point after the new value so that it is interpreted as a decimal number.

```
Option: PATCH
File to patch? <lf>      (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø5
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!2
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ^Z                        (CTRL/Z for new module)
Module name? OVR         (or FIP)
Base address? OPNØST
Offset address? ..DPRT-OPNØST
 Base   Offset  Old       New?
??????  ØØ3264  ØØØØ74  ? n.            (the new default)
??????  ØØ3266  ØØØØØ1  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E Executive
File Processor Patches

Seq 3.5.3 0

RESTRICTING DISK MODES - OPTIONAL FILE PROCESSOR PATCH

The RSTS/E monitor normally allows any user to OPEN a disk file using any mode except 4Ø96 (decimal) (the read regardless mode), and requires that a user have privileges to open a disk file with MODE 4Ø96.

You may wish to allow non-privileged users to use mode 4Ø96. or to restrict one or more of the other disk modes (see the RSTS/E Programming Manual for a complete list). The following procedure allows you to patch the RSTS/E monitor to specify which disk file OPEN modes require privileges. Privileged users and non-privileged users running privileged programs would still have full access to all available modes. Attempts by non-privileged users to use restricted modes would be rejected with no error message.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Decide which disk file modes you wish to make privileged. Consult the RSTS/E Programming Manual for the various modes which are available, and determine the associated MODE values. Compute the sum, n, of the MODE values corresponding to the modes you wish to restrict. If you wish to restrict special update mode, but leave normal update mode unrestricted, use a value of 4 (not 5). If you wish to restrict both update modes, use 4+1. You may type the addition directly to the new value question of PATCH. Be sure to include a decimal point after each number so that it is interpreted in decimal.

3. Bootstrap your system disk and use the PATCH option as shown below. If you made the disk utility code resident when you generated the monitor SIL, answer the 'Module name?' question with "FIP"; otherwise, answer with "OVR" as shown.

```
Option: PATCH
File to patch? <lf>      (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø5
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!4
??????  ØØØØØ2  ??????  ? ^Z              (CTRL/Z for new offset)
Offset address? ^Z                        (CTRL/Z for new base)
Base address? ^Z                          (CTRL/Z for new module)
Module name? OVR         (or FIP)
Base address? OPNØST
Offset address? ..RRNP-OPNØST
 Base   Offset  Old       New?
??????  ØØØ634  Ø1ØØØØ  ? n.              (from step 2)
??????  ØØØ636  ØØ4737  ? ^C              (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E Executive
File Processor Patches

Seq 3.5.4 0

MAKING FIP SYS CALLS PRIVILEGED - OPTIONAL FILE PROCESSOR PATCH

The following FIP SYS calls are normally non-privileged:

-25. File Attribute Read/Write
-17. Change File's Run-Time System Name
+1Ø. Assign a Device
+13. Zero a Device or Account
+14. Read Accounting Data
+15. Indexed Directory Lookup
+17. Wildcard Directory Lookup

It is possible to make one or more of these SYS calls privileged. Doing so will restrict the use of that SYS call to only privileged users and/or privileged programs. Note that the standard CUSP programs assume that the above SYS calls are non-privileged; unexpected results may occur using the CUSPs if one or more of the above are made privileged.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Decide which SYS call code you wish to make privileged. Be sure to include a decimal point after the SYS call code to ensure that it is interpreted in decimal.

3. Bootstrap your system disk and use the PATCH option as shown below.

```
Option: PATCH
File to patch? <lf>      (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø5
Offset address? Ø
 Base   Offset  Old      New?
??????  ØØØØØØ  ??????  ? Q!1Ø
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? $UUOTB
Offset address? n.-UU$MIN*2             (from step 2)
 Base   Offset  Old      New?
??????  ??????  ??????  ? Q!1
??????  ??????  ??????  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3 Seq 3.5.5 0
RSTS/E Executive
File Processor Patches Page 1 of 1

NON-PRIVILEGED ACCESS TO NON-FILE STRUCTURED DISKS - OPTIONAL FILE PROCESSOR PATCH

Normally a non-privileged user is given write access to a non-file structured disk if it was that user's open request which mounted the disk as non-file structured. Certain installations may wish to prohibit non-privileged users from ever getting any access to a non-file structured disk.

PROCEDURE:

1. This is a optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Bootstrap your system disk and use the PATCH option as shown below. If you made the disk utility code resident when you generated the monitor SIL, answer the 'Module name?' question with "FIP"; otherwise, answer with "OVR" as shown.

```
Option: PATCH
File to patch? <lf>     (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø5
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!2Ø
??????  ØØØØØ2  ??????  ? ^Z           (CTRL/Z for new offset)
Offset address? ^Z                     (CTRL/Z for new base)
Base address? ^Z                       (CTRL/Z for new module)
Module name? OVR        (or FIP)
Base address? OPNØST
Offset address? ..NFSP-OPNØST
 Base   Offset  Old       New?
??????  ØØ312  ØØ1422  ? 1416
??????  ØØ314  1Ø1Ø14  ? ^C           (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3 Seq 3.5.6 0
RSTS/E Executive
File Processor Patches Page 1 of 1

NON-PRIVILEGED WRITES TO NON-FILE STRUCTURED DISKS - OPTIONAL FILE PROCESSOR PATCH

Normally a non-privileged user is given write access to a non-file structured disk if it was that user's open request which mounted the disk as non-file structured. Certain installations may wish to prohibit non-privileged users from ever obtaining write access to a non-file structured disk.

PROCEDURE:

1. This is a optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Bootstrap your system disk and use the PATCH option as shown below. If you made the disk utility code resident when you generated the monitor SIL, answer the 'Module name?' question with "FIP"; otherwise, answer with "OVR" as shown.

```
Option: PATCH
File to patch? <lf>     (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø5
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!4Ø
??????  ØØØØØ2  ??????  ? ^Z           (CTRL/Z for new offset)
Offset address? ^Z                     (CTRL/Z for new base)
Base address? ^Z                       (CTRL/Z for new module)
Module name? OVR        (or FIP)
Base address? OPNØST
Offset address? ..NFSW-OPNØST
 Base   Offset  Old       New?
??????  ØØØ44Ø  ØØ1ØØØ  ? 1ØØ1
??????  ØØØ442  ØØ5Ø16  ? ^C           (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E Executive
File Processor Patches

ALLOWING CROSS ACCOUNT CREATIONS - OPTIONAL FILE PROCESSOR PATCH

Normally, a non-privileged job cannot create files in any account other than its own. Some installations may wish to allow a non-privileged job to create files in any account within its project number (i.e., create files in any [x,*] account if the job's account is [x,y]).

This optional patch changes the File Processor's cross account check to allow any job to create files within its project. A non-privileged job still cannot create files in accounts outside of its project number group.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL.

2. Bootstrap your system disk and use the PATCH option as shown below. If you made the disk utility code resident when you generated the monitor SIL, answer the 'Module name?' question with "FIP"; otherwise, answer with "OVR" as shown.

```
Option: PATCH
File to patch? <lf>     (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø5
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!1ØØ
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ^Z                        (CTRL/Z for new module)
Module name? OVR        (or FIP)
Base address? OPNØST
Offset address? ..XGRP-OPNØST
 Base   Offset  Old       New?
??????  ØØ3242  ØØ134Ø  ? 24Ø
??????  ØØ3244  124237  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E Executive
File Processor Notes

WRITE PROTECT AGAINST OWNER FILE PROTECTION CODE HANDLING

The "write protect" file protection code bits (owner, group, and world) mean both write protection and deletion protection. Write protect against owner (Bit 1, value 2) has a special meaning in two kinds of implicit file deletion:

1) When a file is created (e.g. OPEN FOR OUTPUT), any file of the same name is implicitly deleted unless that existent file's protection code says write(delete) protect. Normally, a privileged user has the right to override the delete protect status. But, in this implicit delete case, the write(delete) protect against owner bit is not overridden by privileged users. This ensures that a file marked as non-deletable is not deleted mistakenly. Of course, a privileged user can alter the protection code of any file and, therefore, remove the write protect against owner status.

   Note that explicit deletion of files (e.g. UNSAVE, KILL) will always proceed if the user is privileged.

2) When an account is ZEROed, any file that is marked as write(delete) protected against owner is not deleted. Such files are skipped by the zeroing code; all other files are still deleted. Once again, a privileged user can change the protection code of any file and then delete it or zero the account.

SPECIAL LINE PRINTER DRUMS - OPTIONAL DEVICE DRIVER PATCH

Most DEC LP11 line printer drums contain a character which corresponds to octal code ØØ (the ASCII SPACE character). This character may be a blank, the British Pound Sterling character, or some other optional non-standard character. Normally, this character is irrelevant, since the LP11 is usually wired to print only ASCII codes Ø41 through 137 (octal) for the 64 character set, or Ø41 through 176 for the 96 character set. The SPACE character (ØØ), "non-printing" characters (codes ØØØ to Ø11, Ø13, and Ø16 to Ø37), and character codes greater than 137 (on printers with the 64 character set) all print as spaces. The space is created by suppressing printer action rather than actually having a hammer strike a character position on the printer drum.

By making a simple wiring change to the LP11, the range of printing characters can be extended to include ASCII code ØØ. If the printer drum contains a special character in the ØØ position, sending an ØØ code to the printer will cause that special character to be printed.

In conjunction with this change, you can patch the RSTS/E line printer driver to allow special handling of ASCII ØØ on a unit by unit basis. Normally, the driver sends ØØ codes (SPACEs) unchanged to the printer. If the wiring change is not installed in the printer, this character will appear as a space regardless of the symbol on the line printer drum. The patch allows you to specify an ASCII code which the driver will translate to ØØ before sending it to the printer. The special character will thus be printed. In addition, the driver will translate all ØØ codes to Ø37, which is a non-printing character and appears as a space. Note that translation of ØØ is performed only if the special character translation is enabled.

Ideally, you should choose a character which is not commonly used for other purposes but which can be typed and printed on any standard terminal. This allows you to use any standard editor to insert the special character into text destined for the printer. Remember that the conversion in the printer driver overrides the standard symbol (if any) for the selected code. For example, if you select the code 1Ø1 (the letter A), the special character is printed in place of any A sent to the driver. It would therefore be impossible to print an A.

All normal printing characters have some meaning in RSTS/E or BASIC-PLUS. For example, the & character starts a comment field in the EXTEND mode of BASIC-PLUS, is an abbreviation for account [1,5], and is a special character in RUNOFF source files. The @ character is seldom used in printed material, but RSTS/E uses it as an assignable account designator.

If none of the printable characters is suitable, one of the ASCII control characters (codes ØØ1 to Ø37) may be used. For example, CTRL/F (ØØ6) can be generated with most terminals and will echo and print as ^F if UP ARROW mode is selected with TTYSET. It has no meaning to any standard DEC terminal.

As system manager, you must choose the character which is appropriate for your installation by studying the ASCII code table and considering your users' needs. The procedure for installing the patch is described below.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL that includes line printer support.

2. Decide which line printer unit should have special character handling. If more than one unit is to be patched, repeat the steps below for each unit you wish to modify.

3. Select the character to represent the special printer symbol. If this character is a printable character (e.g. &), specify the new value in the patch by simply typing

   '&

   and replacing the & with the character you have chosen. If the character is a control character, specify its ASCII code as an octal number (or decimal number followed by a decimal point), or as 'x-1ØØ to represent CTRL/x. For example, CTRL/N may be specified as

   16

   or as

   14.

   or as

   'N-1ØØ

   in the patch.

4. Bootstrap your system disk and use the PATCH option as shown below. In this example, the symbol n is used to denote the line printer unit being modified. You should type the actual unit number (between zero and the number of printers -1).

```
Option: PATCH
File to patch? <lf>     (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø7
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!!
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? LPDDDB
Offset address? DDS.LP*n+3Ø             (n is the unit number)
 Base   Offset  Old       New?
??????  ??????  ØØØØØØ  ? (new value from step 3)
??????  ??????  ØØØØØØ  ? ^C            (CTRL/C to exit)

Option:
```

5. Install the hardware modification to the printer. The modification is different for the several printer models, as listed below:

   LP01 (2310) Wiring change required to card cage backplane. Remove wire A3A4-36 and connect this wire through a 1K pullup resistor to +5 Volts.

   LP02 (2410) Change to AR16 Data Register Card. Remove card from slot A3A24. Lift Pin 5 on Z15 and tie Pin 5 to ground.

   LP04 (2470) Wiring change required to card cage backplane. Remove wire from A3A24-04 and connect this wire through a 1K pullup resistor to +5 Volts.

   LP05 (2230) Insert jumper W1 on the Logic Control Board 29-21112.

RSTS/E VØ6C-Ø3
RSTS/E Executive
Device Driver Patches

Seq 3.7.2 0

CHANGING DEFAULT PRINTER FORM LENGTH - OPTIONAL DEVICE DRIVER PATCH

When a line printer is opened, the default hardware form length is set to 66. This is the actual form length for most applications. You can change the default form length by installing the patch given below. The legal form length values are between 7. and 127. inclusive. Note that this patch changes the default form length for all line printers on the system.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL which includes line printer support.

2. Bootstrap your system disk and use the PATCH option as shown below. Be sure you include the decimal point after the new value so that it is interpreted as a decimal number.

```
Option: PATCH
File to patch? <lf>      (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø7
Offset address? Ø
 Base    Offset  Old        New?
??????  ØØØØØØ  ??????  ? Q!2
??????  ØØØØØ2  ??????  ? ^Z              (CTRL/Z for new offset)
Offset address? ^Z                        (CTRL/Z for new base)
Base address? ..LP66
Offset address? Ø
 Base    Offset  Old        New?
??????  ØØØØØØ  ØØØ1Ø2  ? n.              (the new default)
??????  ØØØØØ2  112712  ? ^C              (CTRL/C to exit)

Option:
```

3. If you are using the line printer spooling package, you must change the control file (or files) which start the spoolers. Include the /DFLENGTH:n switch in the command line for each spooler to specify the new default form length.

RSTS/E VØ6C-Ø3
RSTS/E Executive
Device Driver Patches

TU16/TU45/TE45 MAGTAPE DENSITY/PARITY DEFAULT - OPTIONAL DEVICE DRIVER PATCH

The default density and parity settings for TU16, TU45, or TE45 magtape (device MM) under RSTS/E VØ6C-Ø3 are 8ØØ BPI and ODD parity. A program can change the density and/or parity value for either file structured or non-file structured operations by using the MAGTAPE function.

You can change your system's default settings for density and/or parity by installing the patch given below. This is useful if you frequently interchange tapes with other systems which use 16ØØ BPI phase encoded defaults. It is also useful if you want to use the 16ØØ BPI recording mode as your installation's standard.

NOTE: All RSTS/E VØ6C-Ø3 distribution magtapes (including optional languages such as COBOL and FORTRAN) are written at 8ØØ BPI with ODD parity. Do not install this patch until after you have completed the system library build procedures, including any optional software.

PROCEDURE:

1. This is an optional patch to the RSTS/E VØ6C-Ø3 executive. It may be installed in any target monitor SIL which is configured for TU16, TU45, or TE45 magtape.

2. Select the desired magtape density and parity. To use 16ØØ BPI phase encoded mode, specify a new value of Ø in the patch. To return to 8ØØ BPI and ODD parity, specify a new value of 14Ø. Any other value will cause file structured operations to fail.

3. Bootstrap your system disk and use the PATCH option as shown below.

```
Option: PATCH
File to patch? <lf>      (LINE FEED for installed monitor SIL)
Module name? RSTS
Base address? $$Ø3Ø7
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!4
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ..MTDP
Offset address? 1
 Base   Offset  Old       New?
??????  ØØØØØ1     14Ø  ? n             (from step 2)
??????  ØØØØØ2     ???  ? ^C            (CTRL/C to exit)

Option:
```

CR11 CARD READER NOTES

The CR11 card reader normally performs a dark check at column 81 of each card. Cards which have been verified using the IBM 129 punch/verifier or similiar devices will have a verification punch in column 81. This will result in a READ CHECK condition.

There is no way to fix this problem in software since the READ CHECK must be cleared manually at the card reader. There is, however, a hardware fix.

The presence or absence of a wire jumper between pad A and pad B on the "CLOCK" card inside the CR11 reader selects either column Ø or column 81 for the dark check.

Refer to the Documation M2ØØ Card Reader Manual for the clock card schematic and the location of this printed circuit card in the reader.

Jumper A-B

In	Selects dark check at Column Ø
Out	Selects dark check at Column 81

The jumper should be present (in) if cards are read which have verification punches in column 81.

SPECIAL PRINT-USING CHARACTERS - OPTIONAL BASIC-PLUS PATCH

The Print-Using option of BASIC-PLUS provides for floating dollar sign, comma insertion every three digits to the left of the decimal point, and, or course, printing of the decimal point. The $-sign is used for the currency symbol in the United States and Canada, but various other symbols are used for other monetary systems. In England it would be useful to float the Pound Sterling symbol (or whatever ASCII character is used to represent that symbol) instead of the $-sign. In France a floating Franc symbol (or F) would be generally more useful than the floating $-sign. The role of the comma and period are also reversed in France (e.g., 9999999.Ø2 should be printed as F9.999.999,Ø2).

To accommodate these differences in monetary systems, Print-Using is coded to allow substitution of any character for the floating currency symbol, the decimal point character, and the "every three digits" character. These three characters are defined by three words in the BASIC-PLUS Run-Time System. System defaults and several suggestions are shown in the table below:

Location	Usage	Default	England	France
..PUCH+Ø	Floating currency symbol	$ (Ø44)	& (Ø46)	F (1Ø6)
..PUCH+2	Decimal point character	. (Ø56)	. (Ø56)	, (Ø54)
..PUCH+4	Every three digits character	, (Ø54)	, (Ø54)	. (Ø56)

Substitution of different characters, such as those listed under England and France above, is done with a simple patch. Any change will affect both the Print-Using format string and the output produced. For example, if the suggestions under "France" were installed, the following Print-Using statement would result in the output shown:

```
PRINT USING "FF#.######,##", 3673298.Ø2, 4545.2Ø

F3.673.298,Ø2
    F4.545,2Ø
```

Since any change in this area does render the Print-Using documentation slightly incorrect, an appropriate note should be published for users of the system.

PROCEDURE:

1. This is an optional patch to the BASIC-PLUS Run-Time System. It may be installed in any BASIC-PLUS Run-Time System configured with Print-Using.

2. Determine the three special characters you wish to use for the floating currency symbol, the decimal point character, and the "every three digits" character. The chosen characters can be entered as either:

   'x where x is the chosen printable character
   n where n is the octal value of the chosen character
   n. where n is the decimal value of the chosen character

3. Bootstrap your system disk and use the PATCH option as shown below. Answer the 'File to patch?' question with the name of the BASIC-PLUS Run-Time System to be patched.

```
Option: PATCH
File to patch? BASIC.RTS          (BASIC-PLUS Run-Time System name)
Base address? $$Ø4Ø1
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!1
??????  ØØØØØ2  ??????  ? ^Z              (CTRL/Z for new offset)
Offset address? ^Z                        (CTRL/Z for new base)
Base address? ..PUCH
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ØØØØ44  ? 'F              (currency sign from step 2)
??????  ØØØØØ2  ØØØØ56  ? ',             (decimal point from step 2)
??????  ØØØØØ4  ØØØØ54  ? '.             ("every three" from step 2)
??????  ØØØØØ6  ??????  ? ^C             (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E BASIC-PLUS
BASIC-PLUS Patches

Seq 4.1.2 0

DEFAULT SCALE FACTOR - OPTIONAL BASIC-PLUS PATCH

The BASIC-PLUS default SCALE factor may be modified by utilizing the procedure in this article.

PROCEDURE:

1. This is an optional patch to the BASIC-PLUS Run-Time System. It may be installed in any BASIC-PLUS Run-Time System configured with the 4-word, scaled math packages.

2. Determine the new BASIC-PLUS default SCALE factor. This default SCALE factor must be between Ø and 6 inclusive.

3. Bootstrap your system disk and use the PATCH option as shown below. Answer the 'File to patch?' question with the name of the BASIC-PLUS Run-Time System to be patched.

```
Option: PATCH
File to patch? BASIC.RTS        (BASIC-PLUS Run-Time System name)
Base address? $$Ø4Ø1
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!2
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ..SCA.
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ØØØØØØ  ? n             (from step 2)
??????  ØØØØØ2  ??????  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3 Seq 4.1.3 0
RSTS/E BASIC-PLUS
BASIC-PLUS Patches

SCALE FACTOR WARNING MESSAGE - OPTIONAL BASIC-PLUS PATCH

The warning message (%SCALE Factor Interlock) associated with SCALE may be deleted.

PROCEDURE:

1. This is an optional patch to the BASIC-PLUS Run-Time System. It may be installed in any BASIC-PLUS Run-Time System configured with the 4-word, scaled math packages.

2. Bootstrap your system disk and use the PATCH option as shown below. Answer the 'File to patch?' question with the name of the BASIC-PLUS Run-Time System to be patched.

```
Option: PATCH
File to patch? BASIC.RTS        (BASIC-PLUS Run-Time System name)
Base address? $$Ø4Ø1
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!4
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ..SCE.
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  1Ø4577  ? 24Ø           (for no warning message)
??????  ØØØØØ2  ØØØ2Ø7  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E VØ6C-Ø3
RSTS/E BASIC-PLUS
BASIC-PLUS Patches

DEFAULT TO EXTEND MODE - OPTIONAL BASIC-PLUS PATCH

Normally BASIC-PLUS defaults to the NO EXTEND mode. To change this default to EXTEND mode apply the patch in this article.

PROCEDURE:

1. This is an optional patch to the BASIC-PLUS Run-Time System. It may be installed in any BASIC-PLUS Run-Time System.

2. Bootstrap your system disk and use the PATCH option as shown below. Answer the 'File to patch?' question with the name of the BASIC-PLUS Run-Time System to be patched.

```
Option: PATCH
File to patch? BASIC.RTS        (BASIC-PLUS Run-Time System name)
Base address? $$Ø4Ø1
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ??????  ? Q!1Ø
??????  ØØØØØ2  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                      (CTRL/Z for new base)
Base address? ..XTN.
Offset address? Ø
 Base   Offset  Old       New?
??????  ØØØØØØ  ØØØØØØ  ? 2ØØØØ
??????  ØØØØØ2  ??????  ? ^C            (CTRL/C to exit)

Option:
```

The BASIC-PLUS programs supplied by DIGITAL generally assume a default mode of NO EXTEND. Therefore, when building a library or modifying a CUSP, the NO EXTEND mode should be in effect, unless otherwise noted.

RSTS/E V06C-03
RSTS/E BASIC-PLUS
BASIC-PLUS Patches

NO IMMEDIATE MODE FROM .BAS FILE - OPTIONAL BASIC-PLUS PATCH

Normally BASIC-PLUS allows immediate mode commands to appear in the file accessed by the 'OLD' command. To prevent BASIC-PLUS from accepting immediate mode commands in files, apply the patch in this article.

PROCEDURE:

1. This is an optional patch to the BASIC-PLUS Run-Time System. It may be installed in any BASIC-PLUS Run-Time System.

2. Bootstrap your system disk and use the PATCH option as shown below. Answer the 'File to patch?' question with the name of the BASIC-PLUS Run-Time System to be patched.

```
Option: PATCH
File to patch? BASIC.RTS        (BASIC-PLUS Run-Time System name)
Base address? $$0401
Offset address? 0
 Base   Offset  Old       New?
??????  000000  ??????  ? Q!20
??????  000002  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                     (CTRL/Z for new base)
Base address? ..IBAS
Offset address? 0
 Base   Offset  Old       New?
??????  000000  000401  ? 1401
??????  000002  104767  ? ^C            (CTRL/C to exit)

Option:
```

RSTS/E V06C-03 Seq 4.1.6 0
RSTS/E BASIC-PLUS
BASIC-PLUS Patches

NO PPN'S IN CATALOG - OPTIONAL BASIC-PLUS PATCH

Normally BASIC-PLUS allows any user to list the contents of another user's directory with the CATALOG [P,PN] command. The patch in this article prevents BASIC-PLUS from accepting a project, programmer number in the CATALOG command. Thus, only the user's own directory may be listed.

PROCEDURE:

1. This is an optional patch to the BASIC-PLUS Run-Time System. It may be installed in any BASIC-PLUS Run-Time System.

2. Bootstrap your system disk and use the PATCH option as shown below. Answer the 'File to patch?' question with the name of the BASIC-PLUS Run-Time System to be patched.

```
Option: PATCH
File to patch? BASIC.RTS        (BASIC-PLUS Run-Time System name)
Base address? $$0401
Offset address? 0
 Base   Offset  Old       New?
??????  000000  ??????  ? Q!40
??????  000002  ??????  ? ^Z            (CTRL/Z for new offset)
Offset address? ^Z                     (CTRL/Z for new base)
Base address? ..CAT.
Offset address? 0
 Base   Offset  Old       New?
??????  000000  000401  ? 5064
??????  000002  000006  ? ^C            (CTRL/C to exit)

Option:
```

TIME$ FUNCTION

The string produced by the BASIC-PLUS TIME$ function has been upgraded to conform with standardized usage. The string is always 8 characters in length. For 24-hour time, the format is "hh:mm ", where midnight is "00:00 " and noon is "12:00 ". For AM/PM time, the format is "hh:mm xx", where midnight is "12:00 PM" and noon is "12:00 M ".

The following table summarizes:

24-hour	AM/PM	Description
00:00	12:00 PM	Midnight
00:01	12:01 AM	1 minute after midnight
00:59	12:59 AM	59 minutes after midnight
01:00	01:00 AM	1 hour after midnight
11:59	11:59 AM	1 minute before noon
12:00	12:00 M	Noon
12:01	12:01 PM	1 minute after noon
12:59	12:59 PM	59 minutes after noon
13:00	01:00 PM	1 hour after noon
23:59	11:59 PM	1 minute before midnight

According to the Harper Dictionary of Contemporary Usage (Morris, William and Mary, Harper and Row, 1975, p. 30),

> The designations "A.M." for the period between midnight and noon and "P.M." for that between noon and midnight are indisputable and never confused. Some persons are confused, though, as to how to write "12 noon" and "12 midnight."
>
> The key is the letter "M" which appears in "A.M." and "P.M." It is the abbreviation of "meridies", the Latin word for midday. Thus the correct designation for noon is "12 M." (for meridies); midnight is "12 P.M." (for "post meridiem"). Hours between midnight and midday are, of course, "A.M." - "ante meridiem."
>
> Occasionally you will see in print "12 N." for noon and "12 M." for midnight. These abbreviations are wrong.
>
> The forms given are those officially recognized by the U.S. Naval Observatory.

COMMA PRINT ZONE HANDLING

Comma print zones are a convenient method of formatting data in a columnar fashion when the width of the output device is unknown.

Comma print zones are fourteen (14) characters in length. The first comma print zone starts at the left hand margin and succeeding zones start every 14 characters thereafter. If the width of the output device is not a multiple of 14, there are up to 13 character positions not usable in comma print zone processing; partial comma print zones are not allowed.

A comma (,) in a BASIC-PLUS PRINT statement advances the "print head" to the start of the next comma print zone. If, after advancing at least one character position, there would remain a full 14 character comma print zone on the current line, BASIC-PLUS outputs spaces to advance to the start of that zone. Since at least one space is output, adjacent commas in a PRINT statement skip over comma print zones. If no more full comma print zones remain on the current line, BASIC-PLUS advances to the left hand margin of the next line by outputting a carriage return/line feed sequence.

Terminals and Line Printers have a real "line width". For terminals, this width can be set using TTYSET. BASIC-PLUS uses the current width setting for these devices in the comma print zone calculations.

All other devices do not have a "line width". These devices must default "line width" to some reasonable value so that, for example, a file output to disk can later be PIP'd to a terminal or line printer. The default width value used for all "widthless" devices is 72. This corresponds to five (5) comma print zones and ensures that files can be printed on all terminals and line printers without line length problems.

For complete control of output formatting, some other method such as the TAB function or PRINT-USING must be used.

CARRIAGE RETURN HANDLING IN INPUT LINE

This article describes the handling of Carriage Return (ASCII code 13 decimal) by the BASIC-PLUS INPUT LINE statement.

Whenever a Carriage Return (<CR>) is encountered, the next input character is examined. The following three cases exist:

1) <CR> followed by Line Feed (<LF>, ASCII code 1Ø decimal)

   This is the normal ASCII stream line termination case.

   The <CR> and <LF> together serve as the line's delimiter. The string returned by the INPUT LINE ends with <CR><LF>.

2) <CR> followed by Null (<NUL>, ASCII code Ø)

   Both the <CR> and the <NUL> are thrown away; character processing continues as if they never appeared.

   When BASIC-PLUS outputs a Line Feed style continuation line, the line's final three characters are <LF><CR><NUL>. When reading such a file, INPUT LINE will terminate when it detects the <LF> and supply two characters to end the resulting string with <LF><CR><NUL>. The next two input characters are <CR><NUL>, which are ignored at the start of the next INPUT LINE statement.

3) <CR> followed by anything else

   The <CR> is not used as a delimiting character, but is kept and treated as a normal data character.

   This case occurs when overprint lines are being used (e.g., RUNOFF output with underlining). The line's normal form is:

   <data><CR><more data><CR><LF>

   The full <CR><LF> sequence is the line's terminator.

HANDLING OF INTEGER AND FLOATING POINT CONSTANTS

INTEGER CONSTANTS

Versions of BASIC-PLUS prior to VØ6B generated 5 bytes of code and data storage for every reference to an integer constant. These were: 1 byte of pseudo-opcode, 2 bytes of address, and 2 bytes of data (the constant itself). The current version of BASIC-PLUS generates 3 bytes of code and data for all integer constant references: 1 byte of pseudo-opcode and 2 bytes of data (the constant itself). It also treats the integers zero (Ø%) and one (1%) as special cases. These commonly used integers generate only 1 byte of code, a special pseudo-opcode. One programming technique in earlier versions of BASIC-PLUS was to assign some variable permanently to a constant value and use that variable instead of the literal constant throughout the program (variable references generated only 3 bytes of code). With the enhancement of BASIC-PLUS, this technique is no longer necessary and changing a program to use literal integer constants may well save further space!

FLOATING POINT CONSTANTS

Each reference to a floating point constant normally generates 7 (or 11 for 4-word math) bytes of code and data storage: 1 byte of pseudo-opcode, 2 bytes of address, and 4 (or 8) bytes of data (the constant itself). The current version of BASIC-PLUS checks the value of every floating point constant before generating the above sequence of code and data. If the constant has no non-zero bits in its least significant word (or 3 words for 4-word math), BASIC-PLUS generates an alternate 3 byte sequence: 1 byte of special pseudo-opcode and 2 bytes of data (the leading or most significant word of the constant). Most commonly used floating point numbers fall into this special class. Furthermore, BASIC-PLUS will recognize the floating point constants zero (Ø.) and one (1.) and generate only 1 byte of special pseudo-opcode.

INSTALLING AND TROUBLESHOOTING 2780'S

Installing a communication package involves a minimum of two vendors and more likely three or four. The most time-consuming aspect of installation can be pinpointing the problem. When installing A to communicate with B, the problem may be in one of seven areas.

A's software (either code or SYSGEN parameters)
A's hardware
A's modem
phone line
B's modem
B's hardware
B's software

Sometimes a given symptom can have several possible causes. Then it is necessary to go through a step by step elimination procedure.

Here are some ideas and suggestions which may be helpful when difficulties arise in installation of 2780 packages.

1. What is a 2780 and what is its protocol?

For those who are unfamiliar with what a 2780 emulator is emulating, here is a short description of what it is and what its protocol is like.

The 2780 was an unintelligent, hard-wired terminal which could communicate to an IBM host or to another 2780. It read cards and transmitted them over a synchronous link to another computer. It received data back and printed it. The model 2 2780 could also receive back data for a card punch.

The 2780 would send 80 character card image records blocked into a maximum of a 400 character block. For reception it would print or punch a file according to a peripheral selection code (basically an Escape 4 would send a file to the punch). It was able to accept horizontal tabs and would handle them according to the spacing of a special HT record which would be sent at the beginning of the file. The block size was again a maximum of 400 characters.

A synchronous protocol is used to enable the data to be sent back and forth. The protocol determines who speaks, when, and whether or not the other end heard them correctly. When there is synchronous communication, the two modems sample the line at a preset speed (2000 times a second or 4800, etc.), and the data is sent in blocks with one character directly following another. Every message, therefore, whether a single control character or a 400 character block is preceded by four SYNC characters and followed by a PAD. The SYNC characters are used to allow the hardware to tell when valid data is coming (as opposed to noise on the line), and the PAD is used to make sure that the final significant character is fully received. Every data record sent has a CRC (cyclic redundancy check) computed on its characters, and following each record and block (IUS, ETB, and ETX) is a two-character checksum which is used to make sure the data was received correctly.

The major control characters used in the 278Ø protocol are:

ENQ (enquiry):	(1) Used to request permission to send data (bid); (2) Used to request resending of a missed response.
ACKØ (acknowledge):	(1) Used to give permission to send data; (2) Used to accept every other data block.
ACK1 (acknowledge):	Used to accept every other data block.
NAK (negative acknowledge):	Used to reject a data block.
STX (start of text):	Used to indicate the beginning of a data block.
ETB (end of block):	Used to indicate the end of a data block.
ETX (end of text):	Normally used to indicate the end of the final block of the job.
IUS (intermediate record end):	Used to indicate the end of a record.
EOT (end of transmission):	Used to indicate completed transmission and give up control of line.

A sample job might be sent as follows:

```
A                    B
-                    -

ENQ -->                          ;request permission to send

                     <-- ACK0        ;permission given

STX DATA1 ETB -->                ;block of data

                     <-- ACK1        ;acknowledge correct reception

STX DATA2 ETB -->                ;another block

                     <-- ACK0

STX DATA3 ETB -->

                     <-- NAK         ;rejection of data

STX DATA3 ETB -->                ;resending of data block

                     <-- ACK1

STX DATA4 ETX -->

ENQ -->                          ;.3 second silence request for response

                     <-- ACK1        ;response shows did not see block,
                                      since an ACK0 is the expected
                                      response for DATA4

STX DATA4 ETX -->                ;block re-sent

                     <-- ACK0

EOT -->                          ;transmission complete

                     <-- ENQ         ;other side requests permission to
                                      send data
```

2. What to check before installation of the software.

There are some questions to be asked before 278Ø installation.

What operating system will you be communicating with? If it is not another PDP-11, is it one of the usual IBM systems? If it is a different system (e.g., a Univac 11Ø8 with a Comterm front end, an RPG program running on a System 3, etc.) has a demo test been done with the system, and what problems, if any, were encountered?

What kind of modems are being used? If they are not identical, are they compatible. If they are the same type, are the options the same on both? If it is a local connection are modem eliminators being used, or are null modems being used? Null modems require a clocking option on the computer hardware to provide transmit clocking. Are the modems for a speed higher than is warranted by the software? RSX11M, RSX11D, and RSTS/E are not warranted to work over 48ØØ baud.

Make sure that the modem has been grounded on the same ground as the hardware. The communications interface and many modems can be affected by being plugged into a device which is grounded differently.

Is the DP11, DU11, or DUP11 the furthest forward device on the bus (except the system disk) which operates at hardware level 5? If there is a DH11 or DZ11 on the system, it will probably be necessary to raise the hardware level of the communications interface to 6. This is because the DH11 often is programmed to remain at interrupt level until the silo is emptied (the DZ11 has no silo and must interrupt per character).

Have the diagnostics been run for the communications interface and the KG11? The DP, DU, or DUP should be checked out with the turn-around plug at the end of the modem cable or with the modem in maintenance mode.

Two manuals are shipped with the 278Ø software. They are the 278Ø RCS Users Guide* and the 278Ø RCS Installation Notes (DEC-11-CCDNA-A-D). Relevant chapters should be read in advance of installation. There will be some overlap between this article and the manuals, but in many ways they should complement each other.

---

* DEC-11-ORJEA-B-D is for RSTS/E 278Ø only.
DEC-11-CRCSA-A-D, DN1, DN2, DN3, and DN4 are for the other 278Øs.

3. Initial attempt to run

One of the more frustrating circumstances is to bring up the package, issue a transmit command, and have it do absolutely nothing. Silence, however, can be just as informative as a message.

This section will itemize the things that can go wrong in the sequence in which they would be encountered. It is specified when systems react in different ways. The systems involved are RT11, RSX11M, RSX11D, and RSTS/E.

a. Run the program for the first time and
   (1) there is a trap to 4 in RT11 and RSTS/E
   (2) there is an odd address trap abort in RSX11M or D

Probably the device address of the DU11 or DUP11 is incorrect. They are floating device address devices. Check that the device was installed at the correct address (i.e., if the DU11 is the only floating device on the system it will be at 76ØØ4Ø, if the DUP11 is the only floating device it will be at 76ØØ5Ø) and also that the system was told the same address. RSTS/E calculates the address during boot, RT11, RSX11M and D have it specified during taskbuild. The KG11 may also be missing. This is the hardware CRC calculator. If it is missing the software cannot run.

b. Run the program and get ready to make the connect to the other system. Dial the phone, receive the beep from the other end, and push (or pull) the data button and
   (1) the data light does not come on or
   (2) the DSR (or MR) indicator does not light.

Usually a dial-out modem will not give DSR (data set ready) until DTR (data terminal ready) has been presented by the communications interface

   (1) check that the light is not just burned out
   (2) check that you have put the system on-line (all systems but RSTS/E)
   (3) check that the DTR bit is on in the status word
   (4) check that a valid but incorrect device address was not used (RT11, RSX11M and D)
   (5) check that the hardware diagnostics were run, including the one using the modem cable.
   (6) check that the cable is securely plugged into the modem

With a leased line, the modem often has DSR strapped on. In that case, this type of error may not be discovered until later in the procedure (see c. and d.). There are now some dial-out modems (e.g., some MILGO modems) which have DSR strapped on also. They will react the same as the leased line modems.

c. The program has been run and the connection established; a transmit transfer command is issued and a modem (or data set) not ready message is given.
   (1) the data set is not ready (e.g., DSR is not up) or
   (2) a valid, but incorrect, device address has been used.

d. This is at the same stage as c. The first transmit command has been issued, but nothing happens. There are several possible causes.

(1) Incorrect interrupt vector (RSX11M and D). Since in these systems interrupt vector is specified at build time, check that it corresponds to the actual hardware vector.

(2) No response to the bid (RT11, RSX11M and D). These systems do an infinite retry on the bid if they get no response. (RSTS/E does a timeout). Unfortunately, this cause is really a symptom with multiple causes of its own. Some possibilities are:

(a) the remote end software is not up (usually leased line) or the connection was not made (a dial-out modem with DSR strapped on).
(b) the modems have different speeds (e.g., 2ØØØ baud talking to 24ØØ)
(c) the modem is not sending out the signal (this can usually be checked by finding out whether the other side has seen the bid)
(d) the other end is responding, but the turn-around time is too fast (usually on a two-wire dial-out system where there are echo suppressers on the line)
(e) the other end is responding but the modem either does not see it (receive strength too low) or the modem does not pass it back (bad receive leads on the modem), or our hardware does not see it (bad receive chip or modem cable - this should be caught by diagnostics or actual zero turnaround delay (see f.)).

To determine whether or not the bid is actually being sent out, check the receive end to see if it has gone into receive mode. If it is in receive mode, but the transmitting side is still in control mode, the bid is being seen but the response is not (or the response is not considered valid).

e. A transmit command is given and the program aborts. In RT11 if the wrong interrupt vector has been specified the program is aborted and the monitor reprompts. In RSX11M if the 278Ø is running in a partition that can be 'shuffled' the program may have been moved. This will have completely unpredictable effects, since the interrupt will jump to where the correct code used to be.

f. When attempting to transmit there are frequent transmit abort errors. Each system has its own version of the error message, but they all indicate that either eight attempts to send a data block were NAKed, or eight attempts to get a valid response were ignored.

On RSTS/E VØ6B and later releases, a status request will show whether the data is being NAKed or timed-out. In the other packages an error request will show the number of NAKs. If the transmit is aborting, but there are few or no NAKs the problem is timeouts.

If the first block is not being accepted, the KG11 may not be working correctly. If it is improperly seated the program may be sending out a CRC of zeros. This should be easily caught by diagnostics.

On a multi-user system the package should not be run at a higher speed than is warranted.

On a multi-user system with a DH11 or DZ11, the communications interface should be at hardware bus request level 6.

Other possibilities are: modems with different turn-around delays, bad modems, incompatible modems, and noisy phone lines.

If there is an actual zero turn around delay (not just a modem with no delay, but a system with a front-end which is capable of responding without any processing delays) it is possible to have the response return while our hardware is sending trailing pads. Either a minimum delay time should be installed or the number of trailing pads can be decreased.

g. Transmit works fine, but files cannot be received.

Most of the suggestions for f. should be checked. Two of the most likely ones are running at too high a speed for the system, or having the communication interface at level 5 on a system with high DH11 or DZ11 load.

If no data from the other side is accepted at all it is quite possible that the other system is SYSGEN'd to support either a 378Ø or a HASP Workstation instead of a 278Ø. Our data will be acceptable to their side, since the 278Ø is, in effect, a subset of the other two. In both cases, the data block can be too long (they will both send up to 512 characters and the 278Ø can only accept 4ØØ) and the CRC calculations are done on the complete block instead of on each record. The usual indication is that we consistently NAK the very first block that they attempt to send to us.

If the other side is SYSGEN'd as a 278Ø, but is sending too long a record (over 132 data characters plus a two character escape sequence) most of our packages will now accept the record and truncate it (as long as there were no other problems with the data). The user will be informed as to the number of truncated records at the end of the reception.

h. The package has been transmitting and receiving without difficulty. It is in the middle of transmitting and all activity stops. The package is in transmit mode, but nothing is going out.

This is usually caused by losing DSR on a modem which will not give CTS (clear to send) to a RTS (request to send) if DSR has been lost (e.g., BELL 2Ø8B modems). Check the modem, and if this is correct, try to push the talk button and then the data button to try to re-establish the connection. If that doesn't work you will have to redial. RSTS/E VØ6B will timeout in 3Ø seconds and terminate transmit. The other packages will not terminate until operator intervention has taken place.

Installation is complete when the program has been brought up, a connection made, and files transmitted and received.

RSTS/E VØ6C-Ø3
System Utilities Package
SHUTUP

CHANGING 'SHUTUP' SLEEP PARAMETERS - OPTIONAL CUSP PATCH

PROBLEM:

In the "Initial Job Killing Phase", SHUTUP makes two passes through the system job table looking for active jobs which can be logged off the system or KILLed. For those jobs which are attached to a keyboard, the following text string is FORCEd to the terminal's input buffer to simulate the log off sequence:

CTRL/C CTRL/C "BYE/Y" <cr>

Since this method requires the running of the system LOGOUT program, the actual logging off process can consume a fair amount of elapsed time after the text string FORCE is made.

The SHUTUP program is initially set up with the following parameters which control a program pause (SLEEP) at the end of the job table search pass:

```
SLEEP.BASE%      = 1Ø%
SLEEP.INCREMENT% =  2%
```

The length of the SLEEP period in seconds is calculated as:

```
SLEEP time = SLEEP.BASE% + (SLEEP.INCREMENT% * <number of terminals
                                                "killed">)
```

where the number of terminals "killed" is the total number of jobs sent the FORCE text string during the pass through the job table. For example, for 3 terminals, a period of 16 seconds results; for 1Ø terminals, a period of 3Ø seconds results.

Some installations may desire to change these parameters to alter the SLEEP period during this phase of SHUTUP operation. A case in point is when a system has at least one terminal running on the system at 11Ø baud. At this transmission speed, it requires 25-3Ø seconds just to type out the LOGOUT message which appears after typing BYE/Y. Changing the value of SLEEP.BASE% to 3Ø% should eliminate this problem.

SOLUTION:

Below is presented a procedure for changing the value(s) SLEEP.BASE% and/or SLEEP.INCREMENT%.

Assume the following parameter values for the edit session which follows:

```
xx = new SLEEP.BASE%      value
yy = new SLEEP.INCREMENT% value
```

PROCEDURE:

1. For purposes of the discussion, we will assume that the required program to be patched is located on the public disk structure. In the patching procedure, we will refer to the program by its most simple name format: <program name>.BAS. The person performing the patching operation should, if necessary, replace all program references with suitable text according to the requirements of his installation.

   Also, for purposes of presenting the patching procedure, we assume that the patching operation will be performed in a privileged account.

2. To apply the indicated corrections, perform the following RSTS/E system commands.

```
RUN $EDIT<cr>
<EDIT's header line>

#SHUTUP.BAS=SHUTUP.BAS/PATCH<cr>

*H/1031<tab>/V<cr>
1031        SLEEP.BASE% = 10%
*G/10/-2C/xx/V<cr>
1031        SLEEP.BASE% = xx%
*AV<cr>
        \ SLEEP.INCREMENT% = 2%
*G/2/-C/yy/V<cr>
        \ SLEEP.INCREMENT% = yy%
*EX<cr>

#^Z

Ready
```

3. To re-compile the program and re-enter it into the system library type the following RSTS/E commands.

```
OLD SHUTUP<cr>

Ready

COMPILE $SHUTUP<cr>

Ready
```

4. The person making the changes to the program should now take whatever steps are necessary, according to installation guidelines, to save the new version of the program.

5. The source (.BAS) version of the program should now be removed from the public structure by using either the UNSAVE or the KILL commands of RSTS/E.

RSTS/E VØ6C-Ø3
System Utilities Package
SHUTUP

SHUTUP'S RUN-TIME SYSTEM MUST BE THE SYSTEM DEFAULT - SHUTUP RESTRICTION

PROBLEM:

The SHUTUP system program can run to completion only if it is running under the system default Run-Time System. If a RSTS/E system has two BASIC-PLUS Run-Time Systems, the operator must ensure that the SHUTUP program has been compiled under the Run-Time System which is the system default.

HOW TO SAVE FILE ATTRIBUTE DATA AND RUN-TIME SYSTEM NAMES WHEN TRANSFERRING FILES

If you manipulate files which require file attribute information or your RSTS/E system has more than one Run-Time System installed, READ THIS ARTICLE.

FILE ATTRIBUTE DATA

RSTS/E VØ6C provides storage space in a file's UFD entry for extended information about that file. This information (file attribute data) is needed by the RMS-11 file system used by COBOL, BASIC-PLUS-2, and other new languages. Attribute data describe such things as the record structure and file allocation information. File attribute data are required for all files manipulated by the RMS-11 file system.

RUN-TIME SYSTEM NAME

The Run-Time System name tells RSTS/E which of the (possibly) many Run-Time Systems installed is to be associated with an executable program; if the name is wrong or not present, the program cannot execute. It is, therefore, extremely important that the Run-Time System name be preserved in file transfers involving any executable file.

LIMITATIONS OF PIPSML

PIPSML does not retain the Run-Time System name or file attribute information across any file transfer. To restore the Run-Time System name to a file which has been transferred using PIPSML, you must run the UTILTY program (using the NAME command). No program is provided which will simply restore the file attribute data.

LIMITATIONS OF PIPEXT AND PIP.SAV

The PIPEXT and PIP.SAV programs, provided in the SYSGEN kit, retain file attribute and Run-Time System information, but only on disk to disk transfers. PIP.SAV requires the RT11 Run-Time System, which is provided in the SYSGEN system, but not always installed on a target system. For systems which need to move files with file attributes, or which have multiple Run-Time Systems, DIGITAL recommends that PIP.SAV be installed. Installation procedures for PIP.SAV are described in the RSTS/E System Generation Manual. Capabilities of the PIP.SAV program are thoroughly documented in the RSTS/E System User's Guide.

TRANSFERRING FILES ON MAGTAPE

There are three ways to transfer files which have attribute data between systems using magnetic tape:

1. Use the RSTS/E BACKUP Package for RSTS/E-RSTS/E transfers.
2. Use the RMS BACKUP Package for RSX-RSTS/E transfers (when using RMS).
3. Use the FLX program for RSX-RSTS/E transfers (when not using RMS).

Both the standard RSTS/E Backup Package (BACKUP.BAC) and the RMS Backup and Restore utility programs (RMSBCK.TSK and RMSRST.TSK) do preserve file attribute data and Run-Time System names across the backup/restore operations.

No program, other than the two Backup Packages, will retain either file attributes or Run-Time System information for a file when transferring to magtape. If a file is transferred from disk to any non-disk medium, using a PIP-like program, and subsequently transferred back to disk, all attribute and Run-Time System information will be lost. The Run-Time System information may be restored through the NAME command of the UTILTY program. No program is available for restoring the file attribute information.

The program FLX, included in the System Library kit, can be used to transform files from fixed-length or variable-length format to formatted ASCII or formatted binary. In addition, FLX can be used to transform files in formatted ASCII or formatted binary back to variable-length format. This program can be used to interchange files between systems using magnetic tape.

PRINTING RMS FILES

You may use the line printer Spooling Package released with RSTS/E VØ6C to print RMS sequential files. Files longer than the VØ6C limit of 32767 blocks cannot be printed. Note that the Spooling Package cannot be used to print RMS indexed-sequential or relative files.

RSTS/E VØ6C-Ø3
Data Manipulation Package
EDITCH

CHANGING THE EFFECT OF THE POSITION COMMAND - OPTIONAL EDITCH PATCH

PROBLEM:

The Position (nP) command, described in Section 2.2.4 of the RSTS/E Text Editor Manual, functions differently than in RSTS/E VØ6A: it does NOT save the contents of the current buffer before beginning its search.

SOLUTION:

The patching procedure described below will cause the P command to work as follows:

1. The current buffer contents, if any, are output to the primary output device and the buffer is cleared.

2. A page is read into the buffer and searched.

3. If the search is unsuccessful, the page buffer is cleared and step 2 is repeated.

4. If the nth occurrence is found, EDIT returns with the page in which the nth occurrence resides in the buffer, with Dot following the search object.

PROCEDURE:

1. For purposes of the discussion, we will assume that the required program to be patched is located on the public disk structure. In the patching procedure, we will refer to the program by its most simple name format: <program name>.BAS. The person performing the patching operation should, if necessary, replace all program references with suitable text according to the requirements of the installation.

   Also, for purposes of presenting the patching procedure, we assume that the patching operation will be performed in a privileged account.

2. To apply the indicated corrections, perform the following RSTS/E system commands.

```
RUN $EDIT<cr>
<EDIT's header line>

#EDITCH.BAS=EDITCH.BAS/PATCH<cr>

*H/2620<tab>/ØAV<cr>
2620    GOSUB 16ØØØ
*12KV<cr>
27ØØ    IF C%<>78% THEN 28ØØ ELSE       GOSUB 193ØØ
```

```
*I<cr>
2620<tab>GOSUB 19200<lf>
<tab>\ 0%=02%<lf>
<tab>\ IF Z%=0% THEN 9900<lf>
<tab>ELSE<tab>GOSUB 16000<lf>
<tab>\<tab>IF A%=0% THEN 1010<lf>
<tab><tab>ELSE<tab>O%=-1%<lf>
<tab>\<tab><tab>GOTO 2620<lf>
<tab>! OUTPUT THE CURRENT BUFFER; IT IS NOT SEARCHED.<lf>
<tab>! AFTER THE FIRST PASS, LOOP AS FOLLOWS:<lf>
<tab>! READ A PAGE AND SEARCH FOR OCCURRENCES OF THE DESIRED STRING.<lf>
<tab>! IF THE NTH OCCURRENCE IS FOUND (A%=0%) THEN RETURN.<lf>
<tab>! IF UNSUCCESSFUL, CLEAR THE PAGE BUFFER AND REPEAT.<lf>
<cr>
<esc>*V<cr>
2700     IF C%<>78% THEN 2800 ELSE       GOSUB 19300
*EX<cr>
#^Z

Ready
```

3. To re-compile the program and re-enter it into the system library, type the following RSTS/E commands.

```
OLD EDITCH<cr>

Ready

COMPILE $EDITCH<40><cr>

Ready
```

4. The person making the changes to the program should now take whatever steps are necessary, according to installation guidelines, to save the new version of the program.

5. The source (.BAS) version of the program should now be removed from the public structure by using either the UNSAVE or the KILL commands of RSTS/E.

RSTS/E VØ6C-Ø3
Error Control Package
Package Notes

ERRLOG.FIL FILE USED IN VØ6B SHOULD BE DELETED

Because of changes to the ERRLOG.FIL file required due to the increased functionality of the error logging programs, the VØ6B version of ERRLOG.FIL should be deleted prior to the initial running of the VØ6C ERRINT/ERRCPY. This will avoid any conflict that may be caused by inconsistencies between the two versions. ERRCPY will create and initialize a new ERRLOG.FIL file when it determines that the file is not present.

MANUAL SHUTDOWN OF THE LINE-PRINTER SPOOLING PACKAGE

On occasion it may become necessary to shutdown the SPOOL program without shutting down the entire system. There are several ways to perform the shutdown.

At any time, an operator may issue the END command to the spooler. This command conditions the spooler to shut itself down (kill itself) the next time it is ready to get a new job. Thus, if the spooler is printing a job, it will complete the job's output before shutting down. If the spooler is waiting for a job, it will shut down immediately. If the operator doesn't want the job currently printing to complete, the ABORT command may be used to terminate the job.

A second method of shutting down the spooler is through an OFFLINE command. This command causes immediate termination of any job being printed with an ABORT message on the listing. It also causes the END processing, and the spooler will kill itself after the current job.

If the operator wishes to terminate a printing job for shutdown or any other reason, the line printer must be online. Under no circumstances will the spooler consider a job completed until all the data which the spooler has tried to print (including termination messages) has been actually printed. If a job is ABORTed by the operator, the spooler will clear out, at the first opportunity, whatever data has already been buffered; the line printer must be online to allow this. If it is desired that no further output be done, it is necessary to use the UTILTY program to KILL the spooler job. In this case, no cleanup can be performed. This process may be necessary in case of certain hardware failures (for example, if the line printer interface will not raise the READY flag). It should only be used in a crisis. The job which was being printed should be killed from the queue file by a QUE/K command from the operator.

For example, to manually shut down the LPØSPL spooler in logical end mode, the following command is used:

```
PLE/INT LPØSPL:END
```

At the completion of the current job, the spooler will kill itself. To manually shut down the BAØSPL job in immediate mode, the command:

```
PLE/INT BAØSPL:OFFLINE
```

is used. BAØSPL will ABORT the current job, clear it from QUEUE.SYS, and kill itself.

RSTS/E VØ6C-Ø3
Spooling and Operator Services Package
Package Notes

FILES QUEUED WITH WILDCARD SPECIFICATION AND DELETE SWITCH DO NOT GET DELETED

QUEMAN has a restriction in RSTS/E VØ6A, VØ6B, and VØ6C in that files that are queued with wildcard specifications and the delete switch, '/DE', specified are not deleted after processing. This is a design restriction and is intended to remain in effect.

RSTS/E VØ6C-Ø3
Spooling and Operator Services Package
Package Notes

WILDCARD FILE LOOKUP OPERATIONS OF THE SPOOLING PACKAGE - RESTRICTION

It is of importance that one understand the general operation and timing of the file lookup procedure in the spooling package and the restrictions which result.

Assume that a job that was queued with wild card file specifications has progressed to the head of the queue. It will remain there until QUEMAN receives a request for a new job from the spooler.

QUEMAN actually begins the file lookup when the new job request is received. QUEMAN searches the file structure, from the beginning, looking for the first occurrence of a file which matches the file specification. It then sends a file packet, including the file's name, to the spooler. The spooler will then request the next message from QUEMAN.

QUEMAN again starts its search at the beginning, but, this time, looks for the second file matching the file specification. Obviously, in order to find the second file, it must once again encounter the first match. QUEMAN now will either send a file packet, if it finds a match, or an end job packet if it does not. This process is repeated until no match is found.

The potential for problems arises for two reasons: first, the amount of time actually required for the entire lookup process may be minutes. Second, the lookup is sequential, and all future lookups are dependent upon the stability and linear integrity of the preceding lookups.

The consequence of this is that during the file lookup phase there must not be any changes (creations or deletions) to files that match the requested file specification. If there are, it is possible that QUEMAN's relative file numbering scheme will be thrown off, yielding such errors as duplicate files being processed or files being skipped entirely.

Because of this, the following restriction must be enforced when using wild card file specifications:

1. Do not queue files that will not be in a stable existence for the duration of the entire lookup process. If it is necessary to queue a job that depends upon future events use a sufficiently long AFTER switch to ensure that the necessary environment is stable for the lookup.

2. For any job which has been requeued, (i.e., a job which had begun to print and was put back into the queue before completion) those files which match the file specification must not be changed.

RSTS/E VØ6C-Ø3
Operator Services and Spooling Package
Package Notes

NOTES ON THE USE OF THE LA18ØS TERMINAL UNDER RSTS/E VØ6C

The first part of this article describes the behavior of the LA18ØS terminal when its power is turned off, either at system start-up time or at some later time during timesharing, and makes recommendations to prevent loss of data.

If any terminal (including an LA18ØS) is powered off when the system is booted, the device appears to the system to be operational. If a user tries to print a file on the terminal, the system will send characters to the device at the usual speed and assume that they are being printed properly. Obviously, since the device is powered off, the characters are not being printed, but the device can not return any indication of that fact to the system. The entire file will be sent as if it were printing.

If a spooler is started on such a KB: line, any files queued to that spooler will be sent, just as if the device were printing properly. The queue will gradually empty, and any files queued with /DELETE will be deleted.

Since there is no method by which the software can accurately differentiate between a terminal which is powered off and one which is printing properly, it is highly recommended that the person starting up a RSTS/E system check to ensure that all terminal devices which are ordinarily spooled are powered on and properly online before system start-up is performed.

A second class of problems involves the behavior of the LA18ØS when it is powered off or powered on after it has been ASSIGNed or OPENed.

During its power down and power up sequence, the LA18ØS terminal may send spurious characters to the host processor. The ASCII code of these characters appears to be random, but they occasionally take on the value of control characters (for example, CTRL/C). Since the program printing on the terminal may not expect input of any kind from the terminal, these spurious characters may result in unexpected errors: a CTRL/C may cause the program to print READY; a CTRL/O will cause all further output to be discarded without printing; etc. There is no way for the software to protect against all possible spurious inputs. RSTS/E VØ6C is able to protect against all spurious input except CTRL/S and CTRL/Q via the MODE 32% option on the OPEN statement. However, spurious XON/XOFF characters can still cause loss of synchronization between the host system and the terminal. The only way to prevent this problem is to ensure that the LA18ØS is never powered off or powered on while it is ASSIGNed or OPENed.

A final problem concerns loss of data when the LA18ØS is powered off. The LA18ØS contains an internal buffer which is used to hold characters which have been received from the host processor but which have not yet been printed. When the LA18ØS goes OFFLINE (either because the ONLINE/OFFLINE switch is placed in the OFFLINE position, or because some exception condition, such as a paper jam, has been detected) this buffer may still contain some unprinted data. If the terminal is placed back ONLINE, printing will resume with no loss of data. If, however, the terminal is turned off, the buffer is cleared and any data in the buffer is lost. The only protection against this loss is to ensure that the LA18ØS is not powered off while it is printing.

In summary, DIGITAL recommends that any LA18ØS terminal to be used under RSTS/E timesharing be powered on before system start-up and that it remain on whenever it is in use. If the terminal is powered off, loss of data and unexpected errors may result.

Another kind of problem has been encountered when spooling to an LA18ØS. This problem concerns the manner in which the terminal device handles printed lines which are longer than the width of the terminal.

When a line is printed on the LA18ØS, the device prints characters until it gets to the physical right margin. At that point, the logic of the terminal inhibits further printing until a carriage return character moves the print head back from the margin. When a CR is encountered in such a situation, the LA18ØS also generates a line-feed, causing spacing to a new line. Since most print lines are terminated by a CR-LF sequence, however, the second LF will, again, cause spacing to a new line, leaving one line blank. Thus, printing a line which is longer than the width of the LA18ØS (i.e., longer than 132 characters) will cause a blank line on the printed output.

When performing normal print operations under RSTS/E, the user will not ordinarily encounter this problem. The RSTS/E Terminal Service ordinarily keeps track of the characters printed on the terminal and explicitly issues a CR-LF sequence to the terminal when the print head reaches the right margin (as defined by the WIDTH parameter in a TTYSET command). Thus, lines printed to that terminal which are longer than the width of the terminal result in printing of all characters, but on multiple lines. The only programs which will have trouble are those which need to keep track of vertical position on the page (for example, the line printer spooler) and, therefore, invoke the special MODE (Mode 4%) to inhibit the automatic CR-LF by the Terminal Service. Files printed by such programs will have blank lines after each line which was longer than the terminal width. There is no practical software remedy for this problem.

RSTS/E VØ6C-Ø3 Seq 14.6.1 0
Spooling and Operator Services Package
QUMRUN

DISABLING 'JOB WITH DIFFERENT FORM NAME WAITING' MESSAGE - OPTIONAL QUMRUN PATCH

PROBLEM:

QUMRUN will print a message on the OPERATOR SERVICES CONSOLE whenever one of the LP: spoolers has no JOBs which it may print but there are eligible JOBs waiting in the spooler's queue with a form name which is different from the spooler's. It may be desirable to disable this feature.

SOLUTION:

The following patch will permanently disable this feature.

PROCEDURE:

1. For purposes of the discussion, we will assume that the required program to be patched is located on the public disk structure. In the patching procedure, we will refer to the program by its most simple name format: <program name>.BAS. The person performing the patching operation should, if necessary, replace all program references with suitable text according to the requirements of the installation.

   Also, for purposes of presenting the patching procedure, we assume that the patching operation will be performed in a privileged account.

2. To apply the indicated corrections, perform the following RSTS/E system commands.

```
RUN $EDIT<cr>
<EDIT's header line>

#QUMRUN.BAS=QUMRUN.BAS/PATCH<cr>

*H/1380<tab>/V<cr>
1380                        DFORMS% = Ø%
*G/IF Z4% = 2%/I/ AND Ø%/V<cr>
        \       IF Z4% = 2% AND Ø% THEN GOSUB 2410%
*EX<cr>
#^Z

Ready
```

3. To re-compile the program and re-enter it into the system library, type the following RSTS/E commands.

```
OLD QUMRUN<cr>

Ready

COMPILE $QUMRUN<cr>

Ready
```

4. The person making the changes to the program should now take whatever steps are necessary, according to installation guidelines, to save the new version of the program.

5. The source (.BAS) version of the program should now be removed from the public structure by using either the UNSAVE or the KILL commands of RSTS/E.

RSTS/E VØ6C-Ø3
Spooling and Operator Services Package
SPLRUN

Seq 14.1Ø.1 0

ELIMINATING EXTRA FORM FEEDS ON SPECIAL FORMS - OPTIONAL SPLRUN PATCH

PROBLEM:

At the beginning of each job, the spooler performs two top of form operations. The purpose of this action is to ensure that the new job really starts at the top of a new form, with at least one blank page between it and the last job. When using special paper forms, however, it is usually necessary to do a forms alignment. In this case, it may be annoying to have the extra top of form executed because it wastes a page of the form.

SOLUTION:

The following patch inhibits the extra top of form operation at the start of a new job. Note that it inhibits the action not only after a forms alignment process, but also at the beginning of any job. If this patch is used, it is essential that the paper be very carefully aligned in the printer. If the paper is misaligned, one job will begin on the same physical page as the end of the previous job.

PROCEDURE:

1. For purposes of the discussion, we will assume that the required program to be patched is located on the public disk structure. In the patching procedure, we will refer to the program by its most simple name format: <program name>.BAS. The person performing the patching operation should, if necessary, replace all program references with suitable text according to the requirements of the installation. Also, for purposes of presenting the patching procedure, we assume that the patching operation will be performed in a privileged account.

2. To apply the indicated corrections, perform the following RSTS/E system commands.

```
RUN $EDIT<cr>
<EDIT's header line>

#SPLRUN.BAS=SPLRUN.BAS/PATCH<cr>

*H/3200/V<cr>
3200    GOSUB 14500 IF PRTLNE%<>1%
*AV<cr>
        \ NXTLNE%=2% \ RECLEN%=Ø% \ GOSUB 14800
*2KV<cr>
        \ GOSUB 21000
*EX<cr>

#^Z

Ready
```

3. To re-compile the program and re-enter it into the system library, type the following RSTS/E commands.

```
OLD SPLRUN<cr>

Ready

COMPILE $SPLRUN<cr>

Ready
```

4. The person making the changes to the program should now take whatever steps are necessary, according to installation guidelines, to save the new version of the program.

5. The source (.BAS) version of the program should now be removed from the public structure by using either the UNSAVE or the KILL commands of RSTS/E.

FEATURES NOT AVAILABLE ON RSTS/E VØ6C

The following features described in the IAS/RSX-11M RMS-11 MACRO Programmer's Reference Manual (AA-ØØØ2A-TC) are NOT available on RSTS/E VØ6C.

MANUAL SECTION	FEATURE
2.3.2.1	Record Access Streams

Because all record access operations on RSTS/E are synchronous, only one record access stream can be active at any point in time.

5.2.1 Allocation Quantity

Since RSTS/E cannot extend a contiguous file, any file that is declared to be contiguous should be totally allocated when it is created.

5.2.3 Bucket Size

The RSTS/E bucket contains a maximum of 15 blocks.

5.2.11 Default Name String

RSTS/E does not support file version numbers.

5.2.15 File Name String Size

The length of a file specification must conform to RSTS/E conventions; specifically, file-names are no longer than six characters and the file version field is not supported.

5.2.16 FB$CTG Option

If a file is contiguous, it cannot be extended with $EXTEND.

5.2.26 Retrieval Window Size

Since RSTS/E does not support the retrieval window concept, this field is used to define the file's clustersize. The value specified in this field for a $CREATE operation is the number of blocks in each of the file's clusters; the value must be a power of two, though the value 256 is illegal (255 sets the clustersize to 256).

6.1 Allocating a Record Access Block

Only the value SYN for type is meaningful on RSTS/E; ASYN is not used.

MANUAL SECTION FEATURE

6.2.16 Record Processing Options

The option RB$ASY (asynchronous I/O operation request) is not meaningful on RSTS/E.

7.5.3 File Protection

Since file protection is operating system-dependent, this two-byte field must be set to a value valid for the RSTS/E operating system (see the RSTS/E System User's Guide (DEC-11-ORSUB-A-D+DN1)).

7.6 Allocation Extended Attribute Blocks

Placement control is not available on RSTS/E.

7.6.2 Alignment Boundary Type

Since placement control is not available on RSTS/E, the X$ALN field must be set to zero.

7.6.4 Allocation Options
7.6.7 Allocation Starting Point
7.6.8 Volume Number

The capabilities described in these sections are not supported by RSTS/E.

9.2.2 $OPEN

The $OPEN function on RSTS/E does not return a value for either the FB$CTG or the F$RTV field.

9.2.3 $DISPLAY

The $DISPLAY function does not fill in the Date/Time or Protection XABs on RSTS/E.

A.1/A.2 RMS COMPLETION STATUS CODES

The services provided by Files-11 or ACP on RSX-11M are supplied by the operating system in RSTS/E. Therefore, consult the RSTS/E Programming Manual (DEC-11-ORPMA-B-D+DN1), Appendix C, for further explanation of Files-11 and ACP errors. Note that the value returned in STV by RMS-11 when operating on RSTS/E is the negative of the value shown in the manual, that is, if STV contains -2Ø, look up 2Ø.

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Compiler Patches

Seq 27.2.1 M

VIRTUAL ARRAY ELEMENTS MAY BE MISALIGNED

PROBLEM:

The compiler does not always position floating and double precision virtual array elements correctly when there are several arrays in one file.

SOLUTION:

This required patch to the compiler task file will cure the problem.

PROCEDURE:

1. This is a required patch to the compiler task file ($BASIC2.TSK). It must be installed in all versions of the compiler.

2. Obtain a listing of the compiler task build map (BASIC2.MAP). This listing was generated during the build of the compiler when BASIC-PLUS-2 was installed.

3. On the 8th page of the map(SEGMENT: PARSE) at the top, find the line "DISK BLK LIMITS:" and record the first number to the right.

   [DSK] = _____

4. Further down the page, find the follwing symbols and record the first number to the right of each (first column of numbers).

   [. BLK.] = _____

   [$CODE] = _____

   [$IDATA] = _____

5. Install the following patch using the ONLPAT program supplied on the distribution media (where "[symbol]" is to be replaced with the value of the symbol as written above):

```
Command file?  BØ2ØØ1=                    (Produces a permanent record)
File to patch? $BASIC2.TSK
Base address? [DSK]:[$CODE]-[. BLK]
Offset address? 15236
 Base   Offset  Old     New?
??????  Ø15236  163460  ? 163464
```

```
??????  015240  ??????  ? [$IDATA]+210
??????  015242  163562  ? [$IDATA]+206
??????  015244  ??????  ? 162746
??????  015246  163426  ? [$IDATA]+210
??????  015250  ??????  ? 2
??????  015252  162746  ? 160252
??????  015254  ??????  ? [$CODE]+15276
??????  015256  000002  ? 163460
??????  015260  160252  ? [$IDATA]+206
??????  015262  ??????  ? 163562
??????  015264  163460  ? [$IDATA]+20
??????  015266  ??????  ? 163426
??????  015270  163566  ? [$IDATA]+206
??????  015272  000002  ? 160246
??????  015274  163562  ? [$CODE]+15166
??????  015276  ??????  ? 163464
??????  015300  163426  ? [$IDATA]+50
??????  015302  ??????  ? [$IDATA]+206
??????  015304  163460  ? ^C                    (Up-arrow C to exit)
```

6. The compiler is now ready for use.

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Compiler Patches

Seq 27.2.2 M

POSSIBLE FATAL ERRORS ON "PRINT-USING" STATEMENTS

PROBLEM:

The compiler incorrectly compiles PRINT-USING statements without a channel number when the format string is an expression as opposed to a variable or constant. The output when such a statement is executed is random and fatal errors may occur later in the program.

SOLUTION:

This required patch to the compiler task file will cure the problem.

PROCEDURE:

1. This is a required patch to the compiler task file ($BASIC2.TSK). It must be installed in all versions of the compiler.

2. Obtain a listing of the compiler task build map (BASIC2.MAP). This listing was generated during the build of the compiler when BASIC-PLUS-2 was installed.

3. On the 11th page of the map(SEGMENT: IOMOD) at the top, find the line "DISK BLK LIMITS:" and record the first number to the right.

   [DSK] = _____

4. Further down the page, find the follwing symbols and record the first number to the right of each (first column of numbers).

   [. BLK.] = _____

   [$CODE] = _____

   [$IDATA] = _____

   [.$$$$.] = _____

5. Install the following patch using the ONLPAT program supplied on the distribution media (where "[symbol]" is to be replaced with the value of the symbol as written above):

```
Command file?  B02002=                          (Produces a permanent record)
File to patch? $BASIC2.TSK
Base address? [DSK]:[$CODE]-[. BLK]
Offset address? 774
 Base    Offset  Old      New?
??????  000774  163442  ? 163464
??????  000776  000031  ? [.$$$$.]+62
??????  001000  ??????  ? [$IDATA]+12
??????  001002  163464  ? 163442
??????  001004  ??????  ? 2
??????  001006  ??????  ? [$IDATA]+32
??????  001010  163540  ? 160246
??????  001012  ??????  ? [$CODE]+1040
??????  001014  160042  ? 163442
??????  001016  ??????  ? 3
??????  001020  163442  ? [$IDATA]+32
??????  001022  000002  ? 163540
??????  001024  ??????  ? [$IDATA]+12
??????  001026  160246  ? <LF>
??????  001030  ??????  ? <LF>
??????  001032  163442  ? 160246
??????  001034  000003  ? [$CODE]+1014
??????  001036  ??????  ? 160002
??????  001040  163442  ? ^Z                  (Up-arrow Z for new offset)
Offset address? 1054
 Base    Offset  Old      New?
??????  001054  163540  ? 160002
??????  001056  ??????  ? 160002
??????  001060  163460  ? ^C                  (Up-arrow C to exit)
```

6. The compiler is now ready for use.

ERRORS ON ACCESSING VARIABLES WITH DEBUGGER AND "TSK" OUTPUT

PROBLEM:

The compiler incorrectly formats the DEBUG symbol table when TSK output is selected. Therefore, when any attempt to access a variable is made by a user in DEBUG mode, the message "Illegal syntax in PRINT/LET" will result.

SOLUTION:

This required patch to the compiler task file will cure the problem.

PROCEDURE:

1. This is a required patch to the compiler task file ($BASIC2.TSK). It must be installed in all versions of the compiler.

2. Obtain a listing of the compiler task build map (BASIC2.MAP). This listing was generated during the build of the compiler when BASIC-PLUS-2 was installed.

3. On the 29th page(30th for the FPU version) of the map(SEGMENT: OUTPTT) at the top, find the line "DISK BLK LIMITS:" and record the first number to the right.

   [DSK] =
   ----

4. Further down the page, find the follwing symbols and record the first number to the right of each (first column of numbers).

   [. BLK.] =

   [$CODE] =
   ----

5. Install the following patch using the ONLPAT program supplied on the distribution media (where "[symbol]" is to be replaced with the value of the symbol as written above):

```
Command file?  B02003=                          (Produces a permanent record)
File to patch? $BASIC2.TSK
Base address? [DSK]:[$CODE]-[. BLK]
Offset address? 3652
 Base    Offset  Old     New?
??????  003652  ??????  ? Q+224
??????  003654  163464  ? ^Z                    (Up-arrow Z for new offset)
Offset address? 3664
```

```
 Base   Offset  Old     New?
??????  003664  ??????  ? Q+224
??????  003666  163716  ? ^Z                (Up-arrow Z for new offset)
Offset address? 3670
 Base   Offset  Old     New?
??????  003670  ??????  ? Q+224
??????  003672  ??????  ? ^Z                (Up-arrow Z for new offset)
Offset address? 3704
 Base   Offset  Old     New?
??????  003704  ??????  ? Q+224
??????  003706  164200  ? ^Z                (Up-arrow Z for new offset)
Offset address? 4142
 Base   Offset  Old     New?
??????  004142  ??????  ? Q+224
??????  004144  ??????  ? ^C                (Up-arrow C to exit)
```

6. The compiler is now ready for use.

LARGE TERMINAL WIDTHS PRODUCE FATAL ERRORS - MANDATORY BASIC2 COMPILER PATCH

PROBLEM:

If a terminal width of greater than 128. characters is specified prior to executing a BASIC-PLUS-2 program (including the compiler itself), then a MEMORY MANAGEMENT ERROR or a Maximum memory exceeded error will occur.

SOLUTION:

This required patch to the compiler task file will cure the problem.

PROCEDURE:

1. This is a required patch to the compiler task file ($BASIC2.TSK). It must be installed in all versions of the compiler.

2. Obtain a listing of the compiler task build map (BASIC2.MAP). This listing was generated during the build of the compiler when BASIC-PLUS-2 was installed.

3. On the 2nd page of the map(ROOT SEGMENT: BASIC) at the top, find the line "DISK BLK LIMITS:" and record the first number to the right.

   [DSK] =
   ----

4. Further down the page, find the follwing symbol under the column marked "TITLE" and record the first number in the three columns of numbers to the left of the symbol name.

   [$BINIT] =
   ----

5. Install the following patch using the ONLPAT program supplied on the distribution media (where "[symbol]" is to be replaced with the value of the symbol as written above):

```
Command file? B02004=                    (Produces a permanent record)
File to patch? $BASIC2.TSK
Base address? [DSK]:[$BINIT]
Offset address? 142
 Base   Offset  Old     New?
??????  000142  001372  ? 1566
??????  000144  160100  ? ^C             (Up-arrow C to exit)
```

6. The compiler is now ready for use.

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Compiler Patches

Seq 27.2.5 0

POST MORTEM DUMP OPTION ALWAYS ENABLED ON "TSK" OUTPUT

PROBLEM:

The post mortem dump option (/DUMP switch) is always enabled under "TSK" output. Thus a task image dump will always occur on a non-trapped error.

SOLUTION:

This required patch to the compiler task file will cure the problem, making the PMD facility optional.

PROCEDURE:

1. This is a required patch to the compiler task file ($BASIC2.TSK). It must be installed in all versions of the compiler.

2. Obtain a listing of the compiler task build map (BASIC2.MAP). This listing was generated during the build of the compiler when BASIC-PLUS-2 was installed.

3. On the 29th page(3Øth for the FPU version) of the map(SEGMENT: OUTPTT) at the top, find the line "DISK BLK LIMITS:" and record the first number to the right.

   [DSK] =
   ----

4. Further down the page, find the follwing symbols and record the first number to the right of each (first column of numbers).

   [. BLK.] =

   [$CODE] =
   ----

5. Install the following patch using the ONLPAT program supplied on the distribution media (where "[symbol]" is to be replaced with the value of the symbol as written above):

```
Command file?  BØ2ØØ5=                          (Produces a permanent record)
File to patch? $BASIC2.TSK
Base address? [DSK]:[$CODE]-[. BLK]
Offset address? 1Ø456
 Base   Offset  Old     New?
??????  Ø1Ø456  167120  ? 163436
??????  Ø1Ø46Ø  ØØØØØ2  ? 1
??????  Ø1Ø462  164Ø22  ? 163ØØ6
```

```
??????  010464  163460  ? ^Z                (Up-arrow Z for new offset)
Offset address? 10474
 Base   Offset  Old     New?
??????  010474  160234  ? 160252
??????  010476  ??????  ? ^C                (Up-arrow C to exit)
```

6. The compiler is now ready for use.

RSTS/E V06C-03 Seq 27.3.1 M
BASIC-PLUS-2
BASIC2 Run-Time System Patches Page 1 of 2

POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON STRING EQUATES - MANDATORY BASIC2 PATCH

PROBLEM:

If the following conditions are met:
1) a number of files are opened
2) the latest have field statements
3) the files are closed
4) a great deal of string activity takes place
then, it is possible to receive a UNIBUS TIMEOUT fatal trap on a string equate.

SOLUTION:

This required patch to the Run-Time System will cure the problem.

PROCEDURE:

1. This is a required patch to the BASIC2 Run-Time System. It must be installed in all versions of the RTS (EIS, FIS, or FPU).

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                  (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file? B03001=                    (Produces a permanent record)
File to patch? [0,1]BASIC2.RTS
Base address? ROP$
Offset address? 544
 Base   Offset  Old     New?
??????  000544  005711  ? 137
??????  000546  001403  ? PATCH
??????  000550  011000  ? ^Z              (Up-arrow Z for new offset)
Offset address? ^Z                       (Up-arrow Z for new base)
Base address? IFL$
Offset address? 16
 Base   Offset  Old     New?
??????  000016  010061  ? 4737
??????  000020  000024  ? PATCH+30
??????  000022  016060  ? ^Z              (Up-arrow Z for new offset)
Offset address? ^Z                       (Up-arrow Z for new base)
```

```
Base address? PATCH
Offset address? 0
 Base    Offset  Old     New?
??????  000000  000000  ? 32762
??????  000002  000000  ? 10000
??????  000004  000000  ? 220
??????  000006  000000  ? 1004
??????  000010  000000  ? 5711
??????  000012  000000  ? 1404
??????  000014  000000  ? 137
??????  000016  000000  ? ROP$+550
??????  000020  000000  ? 137
??????  000022  000000  ? ROP$+600
??????  000024  000000  ? 137
??????  000026  000000  ? ROP$+556
??????  000030  000000  ? 10061
??????  000032  000000  ? 24
??????  000034  000000  ? 52761
??????  000036  000000  ? 10000
??????  000040  000000  ? 220
??????  000042  000000  ? 207
??????  000044  000000  ? ^C                (Up-arrow C to exit)
```

4. Add the RTS using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                    (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Run Time System Patches

POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON PRINT-USINGS

PROBLEM:

If string expressions are used as the arguments to a PRINT USING statement, it is possible to receive a UNIBUS TIMEOUT fatal trap on that statement.

SOLUTION:

This required patch to the Run Time System will cure the problem.

PROCEDURE:

1. This is a required patch to the BASIC2 Run Time System. It must be installed in all versions of the RTS (EIS, FIS, or FPU).

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                    (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ3ØØ2=                      (Produces a permanent record)
File to patch?  [Ø,1]BASIC2.RTS
Base address? RST$
Offset address? 326
 Base    Offset  Old      New?
??????  ØØØ326  ØØ4767  ? 137Ø3
??????  ØØØ33Ø  ??????  ? 52
??????  ØØØ332  Ø167Ø3  ? 1Ø463
??????  ØØØ334  ??????  ? 224
??????  ØØØ336  Ø1ØØ63  ? 1Ø3Ø4
??????  ØØØ34Ø  ØØØ112  ? 137
??????  ØØØ342  Ø1Ø163  ? PATCH+44
??????  ØØØ344  ØØØ114  ? ^Z                 (Up-arrow Z for new offset)
Offset address? ^Z                           (Up-arrow Z for new base)
Base address? PUS$A
Offset address? 254
 Base    Offset  Old      New?
??????  ØØØ254  ØØ1Ø57  ? 3Ø57
??????  ØØØ256  ØØØØØ6  ? ^Z
Offset address? 4Ø2
 Base    Offset  Old      New?
??????  ØØØ4Ø2  Ø126Ø4  ? 137
??????  ØØØ4Ø4  ØØØ134  ? PATCH+1Ø2
??????  ØØØ4Ø6  126427  ? Z                  (Up-arrow Z for new offset)
```

```
Offset address? 1726
 Base   Offset  Old     New?
??????  001726  000167  ? 137
??????  001730  ??????  ? PATCH+104
??????  001732  010546  ? Z                    (Up-arrow Z for new offset)
Offset address? 2066
 Base   Offset  Old     New?
??????  002066  000167  ? 137
??????  002070  ??????  ? PATCH+104
??????  002072  062706  ? Z                    (Up-arrow Z for new offset)
Offset address? 3434
 Base   Offset  Old     New?
??????  003434  062700  ? <lf>
??????  003436  000112  ? 116
??????  003440  012701  ? <lf>
??????  003442  000012  ? 10
??????  003444  005020  ? Z                    (Up-arrow Z for new offset)
Offset address? Z                            (Up-arrow Z for new base)
Base address? PATCH
Offset address? 44
 Base   Offset  Old     New?
??????  000044  000000  ? 62704
??????  000046  000000  ? 232
??????  000050  000000  ? 12744
??????  000052  000000  ? PATCH+66
??????  000054  000000  ? 10344
??????  000056  000000  ? 62714
??????  000060  000000  ? 112
??????  000062  000000  ? 137
??????  000064  000000  ? MOS$AM
??????  000066  000000  ? 13703
??????  000070  000000  ? 52
??????  000072  000000  ? 16304
??????  000074  000000  ? 224
??????  000076  000000  ? 137
??????  000100  000000  ? RST$+346
??????  000102  000000  ? 12604
??????  000104  000000  ? 4737
??????  000106  000000  ? $MNIUS
??????  000110  000000  ? 134
??????  000112  000000  ? ^C                   (Up-arrow C for exit)
```

4. Add the RTS using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                  (CTRL/Z to exit)
```

RSTS/E V06C-03 Seq 27.3.3 M
BASIC-PLUS-2
BASIC2 Run-Time System Patches Page 1 of 2

CLOSING A WRITE LOCKED OR HUNG DEVICE RESULTS IN A LOOP - MANDATORY BASIC2 PATCH

PROBLEM:

If a previously OPENed device goes off-line and a CLOSE is then attempted, it will possibly fail with a DEVICE WRITE LOCKED OR HUNG error. If there is error trapping in effect and a CLOSE is attempted in the error handler, an endless loop may result.

SOLUTION:

This required patch to the Run-Time System will cure the problem.

PROCEDURE:

1. This is a required patch to the BASIC2 Run-Time System. It must be installed in all versions of the RTS (EIS, FIS, or FPU).

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                   (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  B03003=                          (Produces a permanent record)
File to patch?  [0,1]BASIC2.RTS
Base address? $BOP
Offset address? 514
 Base   Offset  Old     New?
??????  000514  010346  ? 137
??????  000516  016703  ? PATCH+112
??????  000520  ??????  ? 240
??????  000522  016303  ? ^Z                (Up-arrow Z for new offset)
Offset address? Z                          (Up-arrow Z for new base)
Base address? PATCH
Offset address? 112
 Base   Offset  Old     New?
??????  000112  000000  ? 42760
??????  000114  000000  ? 20
??????  000116  000000  ? 136
??????  000120  000000  ? 10346
??????  000122  000000  ? 13703
??????  000124  000000  ? 52
??????  000126  000000  ? 137
??????  000130  000000  ? $BOP+522
??????  000132  000000  ? ^C                (Up-arrow C to exit)
```

4. Add the RTS using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                    (CTRL/Z to exit)
```

RSTS/E V06C-03 Seq 27.3.4 M
BASIC-PLUS-2
BASIC2 Run Time System Patches Page 1 of 1

CHAINING TO A BASIC-PLUS PROGRAM FROM BASIC-PLUS-2 MAY RESULT IN SCALE FACTOR INTERLOCK

PROBLEM:

If a BASIC-PLUS-2 chains to a BASIC-PLUS program, it is possible for the BASIC-PLUS program to give a SCALE FACTOR INTERLOCK error.

SOLUTION:

This required patch to the Run Time System will cure the problem.

PROCEDURE:

1. This is a required patch to the BASIC2 Run Time System. It must be installed in all versions of the RTS (EIS, FIS, or FPU).

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                  (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  B03004=                          (Produces a permanent record)
File to patch?  [0,1]BASIC2.RTS
Base address? ..RUN
Offset address? 406
 Base   Offset  Old     New?
??????  000406  062700  ? <lf>
??????  000410  002764  ? 2766
??????  000412  012046  ? 5046
??????  000414  012046  ? ^C              (Up-arrow C to exit)
```

4. Add the RTS using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                  (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3 Seq 27.3.5 M
BASIC-PLUS-2
BASIC2 Run Time System Patches Page 1 of 1

ERROR TRAPPING OUT OF STRING FUNCTION GIVES MEMORY MANAGEMENT VIOLATION

PROBLEM:

If an error trap . . taken in the middle of a function with string arguments, then it is possible that a MEMORY MANAGEMENT ERROR will occur.

SOLUTION:

This required patch to the Run Time System will cure the problem.

PROCEDURE:

1. This is a required patch to the BASIC2 Run Time System. It must be installed in all versions of the RTS (EIS, FIS, or FPU).

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

*REMOVE BASIC2
*^Z                                  (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ3ØØ5=               (Produces a permanent record)
File to patch?  [Ø,1]BASIC2.RTS
Base address? $GTROM
Offset address? 332
 Base   Offset  Old       New?
??????  ØØØ332  1Ø1ØØ7  ? 1Ø3ØØ7
??????  ØØØ334  Ø16ØØ1  ?  ^C          (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

*ADD BASIC2
*^Z                                  (CTRL/Z to exit)
```

RSTS/E V06C-03 Seq 27.3.6 M
BASIC-PLUS-2
BASIC2 Run-Time System Patches Page 1 of 1

LARGE TERMINAL WIDTHS PRODUCE FATAL ERRORS - MANDATORY BASIC2 PATCH

PROBLEM:

If a terminal width of greater than 128. characters is specified prior to executing a BASIC-PLUS-2 program (including the compiler itself), then a MEMORY MANAGEMENT ERROR or a Maximum memory exceeded error will occur.

SOLUTION:

This required patch to the Run-Time System will cure the problem.

PROCEDURE:

1. This is a required patch to the BASIC2 Run-Time System. It must be installed in all versions of the RTS (EIS, FIS, or FPU).

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                   (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  B03006=                (Produces a permanent record)
File to patch?  [0,1]BASIC2.RTS
Base address? $INITM
Offset address? 142
 Base   Offset  Old       New?
??????  000142  001372  ? 1566
??????  000144  160100  ?  ^C          (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                   (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Run Time System Patches

DEBUG MAY IGNORE BREAK IN USER FUNCTION

PROBLEM:

When using DEBUG, a line which contains a user function call will not BREAK after the function has been executed once.

SOLUTION:

This required patch to the Run Time System will cure the problem.

PROCEDURE:

1. This is a required patch to the BASIC2 Run Time System. It must be installed in only the FPU version of the RTS. For other versions, see sequence 27.5.1.

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                     (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ3ØØ7=                  (Produces a permanent record)
File to patch?  [Ø,1]BASIC2.RTS
Base address? XLN$
Offset address? 4
 Base    Offset  Old      New?
??????  ØØØØØ4  Ø11ØØ2  ? 1Ø41Ø
??????  ØØØØØ6  Ø16412  ? 627 1Ø
??????  ØØØØ1Ø  ØØØØØ2  ? ^C           (Up-arrow C to exit)
```

4. Add the RTS using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                     (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Run-Time System Patches

Seq 27.3.8 M

AN ATTEMPT TO USE A SUBSCRIPT OUT OF RANGE IN DEBUG WILL GIVE MEMORY MANAGEMENT ERRORS - MANDATORY BASIC2 PATCH

PROBLEM:

While in the DEBUGGER, any attempt to access an array element whose subscript is out of range (or, in fact, any normally trappable error) will generate a MEMORY MANAGEMENT VIOLATION.

SOLUTION:

This mandatory patch to the Run-Time System will cure the problem.

PROCEDURE:

1. This is a mandatory patch to the BASIC2 Run-Time System. It should be installed in only the FPU version of the RTS. For other versions, see Article 27.5.2.

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ3ØØ8=                      (Produces a permanent record)
File to patch?  [Ø,1]BASIC2.RTS
Base address? DEBUG$-112
Offset address? 2516
 Base    Offset  Old       New?
161432  ØØ2516  ??????  ? DEBUG$+335Ø
161432  ØØ252Ø  ØØØ11Ø  ? ^Z                 (Up-arrow Z for new offset)
Offset address? 3256
 Base    Offset  Old       New?
161432  ØØ3256  ØØ4767  ? 137ØØ
161432  ØØ326Ø  ??????  ? 52
161432  ØØ3262  ØØØ73Ø  ? 47Ø
161432  ØØ3264  Ø1Ø46Ø  ? ^Z                 (Up-arrow Z for new offset)
Offset address? 3276
 Base    Offset  Old       New?
161432  ØØ3276  ??????  ? DEBUG$+335Ø
161432  ØØ33ØØ  ØØØ11Ø  ? ^Z                 (Up-arrow Z for new offset)
Offset address? 3356
 Base    Offset  Old       New?
161432  ØØ3356  ØØØ432  ? 672
161432  ØØ336Ø  ØØ4767  ? ^Z                 (Up-arrow Z for new offset)
Offset address? 3432
```

```
 Base    Offset  Old        New?
161432  ØØ3432  ??????  ?  DEBUG$+3Ø32
161432  ØØ3434  Ø12704  ?  ^Z                    (Up-arrow Z for new offset)
Offset address? 3444
 Base    Offset  Old        New?
161432  ØØ3444  Ø16700  ?  16ØØ6
161432  ØØ3446  ??????  ?  34Ø
161432  ØØ3450  Ø16ØØ4  ?  5Ø6Ø
161432  ØØ3452  ØØØ342  ?  144
161432  ØØ3454  Ø16ØØ6  ?  4737
161432  ØØ3456  ØØØ34Ø  ?  DEBUG$+1144
161432  ØØ3460  Ø1266Ø  ?  631
161432  ØØ3462  ØØØ11Ø  ?  DEBUG$+314Ø
161432  ØØ3464  ØØØ167  ?  ^C                    (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                             (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Run-Time System Patches

UNQUOTED STRINGS OR "&" AT END OF DATA STATEMENTS CAUSE READ ERRORS

PROBLEM:

If an unqoted string or the character "&" (possibly inserted by the TRANSLATOR) ends a DATA statement, then READs will execute unpredicatably, possibly resulting in DATA format errors or values of zero for integers.

SOLUTION:

This required patch to the Run-Time System will cure the problem.

PROCEDURE:

1. This is a required patch to the BASIC2 Run-Time System. It must be installed in all versions of the RTS (EIS, FIS, or FPU).

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ3ØØ9=                       (Produces a permanent record)
File to patch?  [Ø,1]BASIC2.RTS
Base address? IMF$A
Offset address? 147Ø
 Base    Offset  Old      New?
??????  ØØ147Ø  ØØ53Ø1  ? 466
??????  ØØ1472  122711  ? ^Z                 (Up-arrow Z for next offset)
Offset address? 1512
 Base    Offset  Old      New?
??????  ØØ1512  ØØØ766  ? 455
??????  ØØ1514  ØØ52Ø3  ? ^Z                 (Up-arrow Z for next offset)
Offset address? 1542
 Base    Offset  Old      New?
??????  ØØ1542  ØØ1752  ? 1764
??????  ØØ1544  122713  ? ^Z                 (Up-arrow Z for next offset)
Offset address? 1614
 Base    Offset  Old      New?
??????  ØØ1614  ØØ1424  ? 1526
??????  ØØ1616  Ø32760  ? ^Z                 (Up-arrow Z for next offset)
Offset address? 1646
 Base    Offset  Old      New?
??????  ØØ1646  Ø32760  ? 53Ø1
??????  ØØ165Ø  ØØ4ØØØ  ? 137Ø2
```

```
??????  ØØ1652  ØØØ136  ? 52
??????  ØØ1654  ØØ1ØØ2  ? 262Ø1
??????  ØØ1656  ØØØ167  ? 26
??????  ØØ166Ø  176674  ? 1Ø1255
??????  ØØ1662  ØØØ167  ? 1Ø5711
??????  ØØ1664  176664  ? 1716Ø
??????  ØØ1666  1Ø4413  ? 7Ø1
??????  ØØ17ØØ  Ø057ØØ  ? ^C                    (Up-arrow C to exit)
```

4. Add the RTS using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                             (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Run-Time System Patches

DEVICE "TI:" NOT PROPERLY HANDLED - MANDATORY BASIC2 PATCH

PROBLEM:

The use of device "TI:" for the user terminal (recommended for compatibility with other operating systems) may produce RMS errors dealing with incorrect record sizes or keyboard numbers. This action occurs because of a problem in the emulation of certain RSX directives.

SOLUTION:

This mandatory patch to the Run-Time System will cure the problem, insuring that the directive returns the correct information. NOTE: This patch is the counterpart of Article 27.4.4 for this RTS.

PROCEDURE:

1. This is a mandatory patch to the BASIC2 Run-Time System.

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ3Ø1Ø=                       (Produces a permanent record)
File to patch?  [Ø,1]BASIC2.RTS
Base address? RSXEMU
Offset address? 2676
 Base    Offset  Old        New?
??????  ØØ2676  Ø11423  ? 12423
??????  ØØ27ØØ  Ø22427  ? 12402
??????  ØØ27Ø2  Ø44524  ? 26427
??????  ØØ27Ø4  ØØ1Ø17  ? 177774
??????  ØØ27Ø6  1Ø5Ø67  ? 44524
??????  ØØ271Ø  ??????  ? 1Ø17
??????  ØØ2712  ØØ4567  ? 1Ø5Ø37
??????  ØØ2714  176Ø52  ? 4Ø6
??????  ØØ2716  ØØØØ16  ? 4537
??????  ØØ272Ø  Ø12763  ? ..DOFP
??????  ØØ2722  Ø52124  ? 16
??????  ØØ2724  177776  ? 12763
??????  ØØ2726  116Ø2Ø  ? 52124
??????  ØØ273Ø  ??????  ? 177776
??????  ØØ2732  1ØØØØ1  ? 113Ø2Ø
??????  ØØ2734  ØØ51Ø2  ? 4Ø5
```

```
??????  002736  006202  ? 6202
??????  002740  010223  ? 103001
??????  002742  000401  ? 5102
??????  002744  012423  ? 42702
??????  002746  012423  ? 177600
??????  002750  005023  ? 10223
??????  002752  032767  ? 12423
??????  002754  002000  ? 4737
??????  002756  ??????  ? PATCH+134
??????  002760  001403  ? ^Z                    (Up-arrow Z for new offset)
Offset address? ^Z                             (Up-arrow Z for new base)
Base address? PATCH
Offset address? 134
 Base   Offset  Old       New?
??????  000134  000000  ? 5023
??????  000136  000000  ? 32737
??????  000140  000000  ? 2000
??????  000142  000000  ? 400
??????  000144  000000  ? 207
??????  000146  000000  ? ^C                    (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                            (CTRL/Z to exit)
```

RSTS/E V06C-03
BASIC-PLUS-2
BASIC2 Run-Time System Patches

ILLEGAL FILENAME MAY NOT GIVE ERROR - MANDATORY BASIC2 PATCH

PROBLEM:

If a filename string that is used in an OPEN statement has an illegal character in it, then BASIC-PLUS-2 will terminate the file string scan at that character without giving an error. This can result in strange files being created. For example, use of "LP;" will generate a file "LP." instead of returning an error.

SOLUTION:

This mandatory patch to the Run-Time System will cure the problem.

PROCEDURE:

1. This is a mandatory patch to the BASIC2 Run-Time System.

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  B03011=                       (Produces a permanent record)
File to patch?  [0,1]BASIC2.RTS
Base address? $VREAD
Offset address? 402
 Base    Offset  Old        New?
??????  000402  004767   ? 137
??????  000404  ??????   ? PATCH+146
??????  000406  000207   ? ^Z                (Up-arrow Z for new offset)
Offset address? ^Z                           (Up-arrow Z for new base)
Base address? PATCH
Offset address? 146
 Base    Offset  Old        New?
??????  000146  000000   ? 5737
??????  000150  000000   ? 444
??????  000152  000000   ? 1002
??????  000154  000000   ? 137
??????  000156  000000   ? $MNIUS
??????  000160  000000   ? 104402
??????  000162  000000   ? ^C                (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                    (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Run-Time System Patches

POSSIBLE BAD DATA IF VIRTUAL ARRAY READ CAUSES ERROR - OPTIONAL BASIC2 PATCH

PROBLEM:

When the BASIC-PLUS-2 OTS accesses a virtual array element, it first checks the number of the disk block it needs against a word in the header that holds the number of the block in core. If there is a match, then no disk read is done, if not, then the new block number is inserted in that word and the read attempted. If the read fails and the user, as part of his error routine, resumes back to the original statement, the OTS will find a match and assume the block is in core - even tho the read failed. This means that random garbage will be returned for the value of the element.

SOLUTION:

This optional patch to the Run-Time System will cure the problem, by adding an additional check for validity of the actual read.

PROCEDURE:

1. This is a optional patch to the BASIC2 Run-Time System. It can be installed in any version of the RTS (EIS, FIS, or FPU).

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ3Ø12=                       (Produces a permanent record)
File to patch?  [Ø,1]BASIC2.RTS
Base address? $GTVWR
Offset address? 72
 Base    Offset  Old        New?
??????  ØØØØ72  Ø2Ø361   ? 137
??????  ØØØØ74  177774   ? PATCH+162
??????  ØØØØ76  ØØ15Ø5   ? ^Z                (Up-arrow Z for new offset)
Offset address? 246
 Base    Offset  Old        New?
??????  ØØØ246  ØØ1413   ? 1411
??????  ØØØ25Ø  ??????   ? ^Z                (Up-arrow Z for new offset)
Offset address? 27Ø
 Base    Offset  Old        New?
??????  ØØØ27Ø  Ø42761   ? 24Ø
??????  ØØØ272  ØØØØ2Ø   ? 5Ø61
??????  ØØØ274  177772   ? ^Z                (Up-arrow Z for new offset)
```

```
Offset address? 306
 Base   Offset  Old        New?
??????  000306  004767  ? 137
??????  000310  176072  ? PATCH+210
??????  000312  016703  ? ^Z                    (Up-arrow Z for new offset)
Offset address? ^Z                              (Up-arrow Z for new base)
Base address? PATCH
Offset address? 162
 Base   Offset  Old        New?
??????  000162  000000  ? 20361
??????  000164  000000  ? 177774
??????  000166  000000  ? 1402
??????  000170  000000  ? 137
??????  000172  000000  ? $GTVWR+100
??????  000174  000000  ? 32761
??????  000176  000000  ? 4000
??????  000200  000000  ? 177772
??????  000202  000000  ? 1772
??????  000204  000000  ? 137
??????  000206  000000  ? $GTVWR+312
??????  000210  000000  ? 4737
??????  000212  000000  ? $VREAD
??????  000214  000000  ? 52761
??????  000216  000000  ? 4000
??????  000220  000000  ? 177772
??????  000222  000000  ? 770
??????  000224  000000  ? ^C                    (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                             (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3 Seq 27.3.13 M
BASIC-PLUS-2
BASIC2 Run-Time System Patches Page 1 of 1

PRINT USING IGNORES TRAILING ASTERISK - MANDATORY BASIC2 PATCH

PROBLEM:

If the format string in a PRINT USING statement contains a string having an asterisk or a dollar sign as the final character, then that character will not be printed.

SOLUTION:

This mandatory patch to the Run-Time System will cure the problem.

PROCEDURE:

1. This is a mandatory patch to the BASIC2 Run-Time System. It must be installed in any version of the RTS (EIS, FIS, or FPU).

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BASIC2
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ3Ø13=                       (Produces a permanent record)
File to patch?  [Ø,1]BASIC2.RTS
Base address? PUS$A
Offset address? 2722
 Base   Offset  Old       New?
??????  ØØ2722  Ø05364  ? 5764
??????  ØØ2724  ØØØØØØ  ? ^C                 (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BASIC2
#^Z                                          (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3 Seq 27.4.1 M
BASIC-PLUS-2
BP2COM Run-Time System Patches Page 1 of 1

USE OF BP2COM RTS MAY GIVE TASK BUILDER ERRORS

PROBLEM:

If the BP2COM RTS is employed for user programs, it may give some Task Builder errors of the form "Symbol is ambiguously defined" in module JCON1. This is because the module JCON1 was inadvertently included in the BP2COM RTS.

SOLUTION:

This required patch to the STB file will effectively make the module JCON1 "invisible" to the Task Builder and thus cure the problem.

PROCEDURE:

1. This is a required patch to the BP2COM STB file.

2. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ4ØØ1=                    (Produces a permanent record)
File to patch?  [1,1]BP2COM.STB
Base address? Ø
Offset address? 32Ø
 Base   Offset  Old     New?
??????  ØØØ32Ø  124ØØØ  ? 124ØØ1
??????  ØØØ322  ??????  ? ^Z              (Up-arrow Z for new offset)
Offset address? 376Ø
 Base   Offset  Old     New?
??????  Ø3376Ø  ØØØØØØ  ? 1
??????  Ø3Ø762  ??????  ? ^Z              (Up-arrow Z for new offset)
Offset address? 377Ø
 Base   Offset  Old     New?
??????  Ø3377Ø  Ø34340  ? 34341
??????  Ø3Ø772  ??????  ? ^C              (Up-arrow C to exit)
```

RSTS/E VØ6C-Ø3 Seq 27.4.2 M
BASIC-PLUS-2
BP2COM Run-Time System Patches Page 1 of 2

POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON STRING EQUATES - MANDATORY BP2COM PATCH

PROBLEM:

If the following conditions are met:
1) a number of files are opened
2) the latest have field statements
3) the files are closed
4) a great deal of string activity takes place
then, it is possible to receive a UNIBUS TIMEOUT fatal trap on a string equate.

SOLUTION:

This required patch to the Run-Time System will cure the problem. NOTE: This patch is a partial duplicate of Article 27.3.1; it must be installed in conjunction with Article 27.6.1.

PROCEDURE:

1. This is a required patch to the BP2COM Run-Time System.

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BP2COM
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file? BØ4ØØ2=                       (Produces a permanent record)
File to patch? [Ø,1]BP2COM.RTS
Base address? ROP$
Offset address? 544
 Base   Offset  Old     New?
??????  ØØØ544  Ø05711  ? 137
??????  ØØØ546  ØØ1403  ? PATCH
??????  ØØØ55Ø  Ø11ØØØ  ? ^Z                 (Up-arrow Z for new offset)
Offset address? ^Z                           (Up-arrow Z for new base)
Base address? PATCH
Offset address? Ø
 Base   Offset  Old     New?
??????  ØØØØØØ  ØØØØØØ  ? 32762
??????  ØØØØØ2  ØØØØØØ  ? 1ØØØØ
??????  ØØØØØ4  ØØØØØØ  ? 22Ø
??????  ØØØØØ6  ØØØØØØ  ? 1ØØ4
??????  ØØØØ1Ø  ØØØØØØ  ? 5711
??????  ØØØØ12  ØØØØØØ  ? 14Ø4
??????  ØØØØ14  ØØØØØØ  ? 137
```

```
??????  000016  000000  ? ROP$+550
??????  000020  000000  ? 137
??????  000022  000000  ? ROP$+600
??????  000024  000000  ? 137
??????  000026  000000  ? ROP$+556
??????  000030  000000  ? ^C                (Up-arrow C to exit)
```

4. Add the RTS using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BP2COM
#^Z                                  (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3 Seq 27.4.3 M
BASIC-PLUS-2
BP2COM Run-Time System Patches Page 1 of 1

ERROR TRAPPING OUT OF STRING FUNCTION GIVES MEMORY MANAGEMENT VIOLATION

PROBLEM:

If an error trap is taken in the middle of a function with string arguments, then it is possible that a MEMORY MANAGEMENT ERROR will occur.

SOLUTION:

This required patch to the Run-Time System will cure the problem. NOTE: this patch is the counterpart of Art. 27.3.5 for this RTS.

PROCEDURE:

1. This is a required patch to the BP2COM Run-Time System.

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BP2COM
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ4ØØ3=                       (Produces a permanent record)
File to patch?  [Ø,1]BP2COM.RTS
Base address? $GTROM
Offset address? 332
 Base   Offset  Old       New?
??????  ØØØ332  1Ø1ØØ7  ? 1Ø3ØØ7
??????  ØØØ334  Ø16ØØ1  ?  ^C                 (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BP2COM
#^Z                                          (CTRL/Z to exit)
```

RSTS/E V06C-03 Seq 27.4.4 M
BASIC-PLUS-2
BP2COM Run-Time System Patches Page 1 of 2

DEVICE "TI:" NOT PROPERLY HANDLED - MANDATORY BP2COM PATCH

PROBLEM:

The use of device "TI:" for the user terminal (recommended for compatibility with other operating systems) may produce RMS errors dealing with incorrect record sizes or keyboard numbers. This action occurs because of a problem in the emulation of certain RSX directives.

SOLUTION:

This mandatory patch to the Run-Time System will cure the problem, insuring that the directive returns the correct information. NOTE: This patch is the counterpart of Article 27.3.10 for this RTS.

PROCEDURE:

1. This is a mandatory patch to the BP2COM Run-Time System.

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BP2COM
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  B04004=                       (Produces a permanent record)
File to patch?  [0,1]BP2COM.RTS
Base address? RSXEMU
Offset address? 2676
 Base   Offset  Old       New?
??????  002676  011423  ? 12423
??????  002700  022427  ? 12402
??????  002702  044524  ? 26427
??????  002704  001017  ? 177774
??????  002706  105067  ? 44524
??????  002710  ??????  ? 1017
??????  002712  004567  ? 105037
??????  002714  176052  ? 406
??????  002716  000016  ? 4537
??????  002720  012763  ? ..DOFP
??????  002722  052124  ? 16
??????  002724  177776  ? 12763
??????  002726  116702  ? 52124
??????  002730  ??????  ? 177776
??????  002732  100001  ? 113702
??????  002734  005102  ? 405
```

```
??????  002736  006202  ? 6202
??????  002740  010223  ? 103001
??????  002742  000401  ? 5102
??????  002744  012423  ? 42702
??????  002746  012423  ? 177600
??????  002750  005023  ? 10223
??????  002752  032767  ? 12423
??????  002754  002000  ? 4737
??????  002756  ??????  ? PATCH+30
??????  002760  001403  ? ^Z                    (Up-arrow Z for new offset)
offset address? ^Z                             (Up-arrow Z for new base)
Base address? PATCH
Offset address? 30
 Base   Offset  Old       New?
??????  000030  000000  ? 5023
??????  000032  000000  ? 32737
??????  000034  000000  ? 2000
??????  000036  000000  ? 400
??????  000040  000000  ? 207
??????  000042  000000  ? ^C                    (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BP2COM
#^Z                                            (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3 Seq 27.4.5 0
BASIC-PLUS-2
BP2COM Run-Time System Patches Page 1 of 1

CHANGING PROMPT FOR BP2COM WHEN DEFAULT RUN-TIME SYSTEM - OPTIONAL BP2COM PATCH

PROBLEM:

As distributed, the BP2COM RTS prompts with "Ready" even when it is the user default Run-Time System. This may tend to confuse users who expect the "Ready" prompt only from the BASIC-PLUS RTS.

SOLUTION:

This optional patch to the Run-Time System will change the prompt message to "Basic2". Other prompts can be substituted as the user wishes, simply by changing the proper letters in the patch.

PROCEDURE:

1. This is a optional patch to the BP2COM Run-Time System.

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BP2COM
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ4ØØ5=                       (Produces a permanent record)
File to patch?  [Ø,1]BP2COM.RTS
Base address? PROMPT
Offset address? Ø
 Base   Offset  Old       New?
176466  ØØØØØØ  ??????  ? "BA+2ØØØØ
176466  ØØØØØ2  ??????  ? "SI+2ØØ4Ø
176466  ØØØØØ4  ??????  ? "C2+4Ø
176466  ØØØØØ6  ØØ5Ø15  ? ^C                  (Up-arrow C to exit)
```

4. Add the RTS back using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BP2COM
#^Z                                          (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3 Seq 27.4.6 M
BASIC-PLUS-2
BP2COM Run-Time System Patches Page 1 of 1

CHAINING TO A BASIC-PLUS PROGRAM FROM BASIC-PLUS-2 MAY RESULT IN SCALE FACTOR INTERLOCK - MANDATORY BP2COM PATCH

PROBLEM:

If a BASIC-PLUS-2 chains to a BASIC-PLUS program, it is possible for the BASIC-PLUS program to give a SCALE FACTOR INTERLOCK error.

SOLUTION:

This required patch to the Run-Time System will cure the problem. NOTE: This patch is the counterpart of Article 27.3.4 for this RTS.

PROCEDURE:

1. This is a required patch to the BP2COM Run-Time System.

2. The RTS must be removed with the UTILTY program as follows:

```
RUN $UTILTY

#REMOVE BP2COM
#^Z                                          (CTRL/Z to exit)
```

3. Install the following patch using the ONLPAT program supplied on the distribution media:

```
Command file?  BØ4ØØ6=                       (Produces a permanent record)
File to patch?  [Ø,1]BASIC2.RTS
Base address? ..RUN
Offset address? 4Ø6
 Base   Offset  Old     New?
??????  ØØØ4Ø6  Ø62ØØ  ? <1f>
??????  ØØØ41Ø  ØØ276Ø  ? 2766
??????  ØØØ412  Ø12Ø46  ? 5Ø46
??????  ØØØ414  Ø12Ø46  ? ^C                  (Up-arrow C to exit)
```

4. Add the RTS using the UTILTY program as follows:

```
RUN $UTILTY

#ADD BP2COM
#^Z                                          (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3 Seq 27.5.1 M
BASIC-PLUS-2
BASIC2 Object Library Patches Page 1 of 1

DEBUG MAY IGNORE BREAK IN USER FUNCTION

PROBLEM:

When using DEBUG, a line which contains a user function call will not BREAK after the function has been executed once.

SOLUTION:

This required patch to the object library file (BASIC2.OLB) will cure the problem.

PROCEDURE:

1. Build the following correction file using an editor and name it: DEBUG.COR

```
.TITLE  $DEBUG
.IDENT  /Ø7REA/
.PSECT  $DEBUG
.=.+364
MOV     R4,@RØ
ADD     #2,@RØ
.END
```

2. Assemble the correction file as follows:

```
RUN $MAC.TSK
MAC>DEBUG.PAT,DEBUG.LIS=DEBUG.COR
MAC>^Z                          (CTRL/Z to exit)
```

3. Extract the original module as follows:

```
RUN $LBR.TSK
LBR>DEBUG.OLD=[1,1]BASIC2/EX:$DEBUG
LBR>^Z                          (CTRL/Z to exit)
```

4. Run the patch utility to merge old module and correction:

```
RUN $PAT.TSK
PAT>DEBUG=DEBUG.OLD/CS:166557,DEBUG.PAT/CS:7413
PAT>^Z                          (CTRL/Z to exit)
```

5. Replace patched module in library:

```
RUN $LBR.TSK
LBR>[1,1]BASIC2/RP=DEBUG
LBR>^Z                          (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASIC2 Object Library Patches

AN ATTEMPT TO USE A SUBSCRIPT OUT OF RANGE IN DEBUG WILL GIVE MEMORY MANAGEMENT ERRORS - MANDATORY BASIC2 PATCH

PROBLEM:

While in the DEBUGGER, any attempt to access an array element whose subscript is out of range (or, in fact, any normally trappable error) will generate a MEMORY MANAGEMENT VIOLATION.

SOLUTION:

This required patch to the object library file (BASIC2.OLB) will cure the problem.
NOTE: The patch in Article 27.5.1 must have been previously installed.

PROCEDURE:

1. Build the following correction file using an editor and name it: DEBUG.COR

```
        .TITLE  $DEBUG
        .IDENT  /Ø7REB/
        .PSECT  $DEBUG
        $$PC=.
.=$$PC+2514
        MOV     #LPERPT,11Ø(RØ)
.=$$PC+3144
2$:
.=$$PC+3256
3$:     MOV     @#52,RØ
        BR      1$
.=$$PC+3274
        MOV     #LPERPT,11Ø(RØ)
.=$$PC+3356
        BR      2$
.=$$PC+343Ø
        MOV     #2$,@R4
.=$$PC+3444
1$:     MOV     34Ø(RØ),SP
        CLR     144(RØ)
        JSR     PC,$$PC+1256
        BR      2$
LPERPT: $$PC+3252
.END
```

2. Assemble the correction file as follows:

```
RUN $MAC.TSK
MAC>DEBUG.PAT,DEBUG.LIS=DEBUG.COR
MAC>^Z                          (CTRL/Z to exit)
```

3. Extract the original module as follows:

```
RUN $LBR.TSK
LBR>DEBUG.OLD=[1,1]BASIC2/EX:$DEBUG
LBR>^Z                              (CTRL/Z to exit)
```

4. Run the patch utility to merge old module and correction:

```
RUN $PAT.TSK
PAT>DEBUG=DEBUG.OLD/CS:171271,DEBUG.PAT/CS:17400
PAT>^Z                              (CTRL/Z to exit)
```

5. Replace patched module in library:

```
RUN $LBR.TSK
LBR>[1,1]BASIC2/RP=DEBUG
LBR>^Z                              (CTRL/Z to exit)
```

RSTS/E VØ6C-Ø3 Seq 27.6.1 M
BASIC-PLUS-2
BP2COM Object Library Patches Page 1 of 2

POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON STRING EQUATES - MANDATORY BP2COM PATCH

PROBLEM:

If the following conditions are met:
1) a number of files are opened
2) the latest have field statements
3) the files are closed
4) a great deal of string activity takes place
then, it is possible to receive a UNIBUS TIMEOUT fatal trap on a string equate.

SOLUTION:

This required patch to the object library file (BP2COM.OLB) will cure the problem. NOTE: This patch is the object libray counterpart ARTICLE 27.3.1. It must be installed in conjunction with Article 27.4.2.

PROCEDURE:

1. Build the following correction file using an editor and name it: IEFLD.COR

```
        .TITLE  $IEFLD
        .IDENT  /Ø3REA/
        .PSECT
        $$PC=.
        .=$$PC+16
        BR      PATCH
        NOP
XXX:
        .=$$PC+17Ø
PATCH:  MOV     RØ,24(R1)
        BIS     #1ØØØØ,22Ø(R1)
        BR      XXX
        .END
```

2. Build the following correction file using an editor and name it: ICROP.COR

```
        .TITLE  $ICROP
        .IDENT  /25CMA/
        .PSECT
        $$PC=.
        .=$$PC+544
        BR      PATCH
        NOP
XXX:
        .=$$PC+556
YYY:
        .=$$PC+6ØØ
RT:
```

```
        .=$$PC+666
PATCH:  BIT     #10000,220(R2)
        BNE     RT
        TST     @R1
        BEQ     YYY
        BR      XXX
        .END
```

3. Assemble the correction files as follows:

```
RUN $MAC.TSK
MAC>IEFLD.PAT,IEFLD.LIS=IEFLD.COR
MAC>ICROP.PAT,ICROP.LIS=ICROP.COR
MAC>^Z                          (CTRL/Z to exit)
```

4. Extract the original modules as follows:

```
RUN $LBR.TSK
LBR>IEFLD.OLD=[1,1]BP2COM/EX:$IEFLD
LBR>ICROP.OLD=[1,1]BP2COM/EX:$ICROP
LBR>^Z                          (CTRL/Z to exit)
```

5. Run the patch utility to merge old modules and corrections:

```
RUN $PAT.TSK
PAT>IEFLD=IEFLD.OLD/CS:25425,IEFLD.PAT/CS:6047
PAT>ICROP=ICROP.OLD/CS:124011,ICROP.PAT/CS:10174
PAT>^Z                          (CTRL/Z to exit)
```

6. Replace patched modules in library:

```
RUN $LBR.TSK
LBR>[1,1]BP2COM/RP=ICROP,IEFLD
LBR>^Z                          (CTRL/Z to exit)
```

7. Re-TASK BUILD any program that used these modules.

RSTS/E VØ6C-Ø3 Seq 27.6.2 M
BASIC-PLUS-2
BP2COM Object Library Patches Page 1 of 2

POSSIBLE UNIBUS TIMEOUT FATAL ERRORS ON PRINT-USINGS - MANDATORY BP2COM PATCH

PROBLEM:

If string expressions are used as the arguments to a PRINT USING statement, it is possible to receive a UNIBUS TIMEOUT fatal trap on that statement.

SOLUTION:

This required patch to the object library file (BP2COM.OLB) will cure the problem.

PROCEDURE:

1. Build the following correction file using an editor and name it: ICINI.COR

```
        .TITLE  $ICINI
        .IDENT  /Ø5CMA/
        .ENABL  AMA
        .PSECT
        .GLOBL  MOS$AM
        $$PC=.
        .=$$PC+326
        MOV     @#52,R3
        MOV     R4,224(R3)
        MOV     R3,R4
        JMP     PATCH
        NOP
XXX:
        .=$$PC+512
PATCH:  ADD     #232,R4
        MOV     #3$,-(R4)
        MOV     R3,-(R4)
        ADD     #112,@R4
        JMP     MOS$AM
3$:     MOV     @#52,R3
        MOV     224(R3),R4
        JMP     XXX
        .END
```

2. Build the following correction file using an editor and name it: ICPRG.COR

```
        .TITLE  $ICPRG
        .IDENT  /15CMA/
        .ENABL  AMA
        .PSECT
        .GLOBL  $MNIUS
        $$PC=.
        .=$$PC+254
```

```
            .WORD   3057
            .=$$PC+402
            JMP     PATCH
            .=$$PC+1726
            JMP     PATCH+2
            .=$$PC+2066
            JMP     PATCH+2
            .=$$PC+3434
            ADD     #116,R0
            MOV     #10,R1
            .=$$PC+3454
PATCH:      MOV     (SP)+,R4
            JSR     PC,$MNIUS
            JMP     @(R4)+
            .END
```

3. Assemble the correction files as follows:

```
RUN $MAC.TSK
MAC>ICINI.PAT,ICINI.LIS=ICINI.COR
MAC>ICPRG.PAT,ICPRG.LIS=ICPRG.COR
MAC>^Z                          (CTRL/Z to exit)
```

4. Extract the original modules as follows:

```
RUN $LBR.TSK
LBR>ICINI.OLD=[1,1]BP2COM/EX:$ICINI
LBR>ICPRG.OLD=[1,1]BP2COM/EX:$ICPRG
LBR>^Z                          (CTRL/Z to exit)
```

5. Run the patch utility to merge old module¬ and corrections:

```
RUN $PAT.TSK
PAT>ICINI=ICINI.OLD/CS:71565,ICINI.PAT/CS:17401
PAT>ICPRG=ICPRG.OLD/CS:105743,ICPRG.PAT/CS:13636
PAT>^Z                          (CTRL/Z to exit)
```

6. Replace patched modules in library:

```
RUN $LBR.TSK
LBR>[1,1]BP2COM/RP=ICINI,ICPRG
LBR>^Z                          (CTRL/Z to exit)
```

7. Re-TASK BUILD any programs that used these modules.

RSTS/E VØ6C-Ø3 Seq 27.6.3 M
BASIC-PLUS-2
BP2COM Object Library Patches Page 1 of 2

CLOSING A WRITE LOCKED OR HUNG DEVICE RESULTS IN A LOOP - MANDATORY BP2COM PATCH

PROBLEM:

If a previously OPENed device goes off-line and a CLOSE is then attempted, it will possibly fail with a DEVICE WRITE LOCKED OR HUNG error. If there is error trapping in effect and a CLOSE is attempted in the error handler, an endless loop may result.

SOLUTION:

This required patch to the object library file (BP2COM.OLB) will cure the problem. This patch is the counterpart to Article 27.3.3 for this component.

PROCEDURE:

1. Build the following correction file using an editor and name it: IVOPN.COR

```
        .TITLE  $IVOPN
        .IDENT  /21REA/
        .PSECT
        .ENABL  AMA
        $$PC=.
        .=$$PC+514
        JMP     PATCH
        NOP
XXX:
        .=$$PC+710
PATCH:  BIC     #20,136(R0)
        MOV     R3,-(SP)
        MOV     @#52,R3
        JMP     XXX
        .END
```

2. Assemble the correction file as follows:

```
RUN $MAC.TSK
MAC>IVOPN.PAT,IVOPN.LIS=IVOPN.COR
MAC>^Z                          (CTRL/Z to exit)
```

3. Extract the original module as follows:

```
RUN $LBR.TSK
LBR>IVOPN.OLD=[1,1]BP2COM/EX:$IVOPN
LBR>^Z                          (CTRL/Z to exit)
```

4. Run the patch utility to merge old module and correction:

```
RUN $PAT.TSK
PAT>IVOPN=IVOPN.OLD/CS:113442,IVOPN.PAT/CS:1Ø6Ø5
PAT>^Z                              (CTRL/Z to exit)
```

5. Replace patched module in library:

```
RUN $LBR.TSK
LBR>[1,1]BP2COM/RP=IVOPN
LBR>^Z                              (CTRL/Z to exit)
```

6. Re-TASK BUILD any programs that used this module.

RSTS/E V06C-03 Seq 27.6.4 M
BASIC-PLUS-2
BP2COM Object Library Patches Page 1 of 1

ERROR TRAPPING OUT OF STRING FUNCTION GIVES MEMORY MANAGEMENT VIOLATION

PROBLEM:

If an error trap is taken in the middle of a function with string arguments, then it is possible that a MEMORY MANAGEMENT ERROR will occur.

SOLUTION:

This required patch to the object library file (BP2COM.OLB) will cure the problem. This patch is the counterpart to Article 27.3.5 for this component.

PROCEDURE:

1. Build the following correction file using an editor and name it: STMSC.COR

```
.TITLE  $STMSC
.IDENT  /05CMA/
.PSECT
.=.+332
BHIS    .+20
.END
```

2. Assemble the correction file as follows:

```
RUN $MAC.TSK
MAC>STMSC.PAT,STMSC.LIS=STMSC.COR
MAC>^Z                          (CTRL/Z to exit)
```

3. Extract the original module as follows:

```
RUN $LBR.TSK
LBR>STMSC.OLD=[1,1]BP2COM/EX:$STMSC
LBR>^Z                          (CTRL/Z to exit)
```

4. Run the patch utility to merge old module and correction:

```
RUN $PAT.TSK
PAT>STMSC=STMSC.OLD/CS:106747,STMSC.PAT/CS:5344
PAT>^Z                          (CTRL/Z to exit)
```

5. Replace patched module in library:

```
RUN $LBR.TSK
LBR>[1,1]BP2COM/RP=STMSC
LBR>^Z                          (CTRL/Z to exit)
```

6. Re-TASK BUILD any programs that used this module.

RSTS/E VØ6C-Ø3 Seq 27.6.5 M
BASIC-PLUS-2
BP2COM Object Library Patches Page 1 of 2

LARGE TERMINAL WIDTHS PRODUCE FATAL ERRORS - MANDATORY BP2COM PATCH

PROBLEM:

If a terminal width of greater than 128. characters is specified prior to executing a BASIC-PLUS-2 program (including the compiler itself), then a MEMORY MANAGEMENT ERROR or a Maximum memory exceeded error will occur.

SOLUTION:

This required patch to the object library file (BP2COM.OLB) will cure the problem. This patch is the counterpart to Article 27.3.6 for this library.

PROCEDURE:

1. Build the following correction file using an editor and name it: BINIT.COR

```
.TITLE  $BINIT
.IDENT  /27REA/
.PSECT
.=.+142
.WORD   1566
.END
```

2. Assemble the correction file as follows:

```
RUN $MAC.TSK
MAC>BINIT.PAT,BINIT.LIS=BINIT.COR
MAC>^Z                          (CTRL/Z to exit)
```

3. Extract the original module as follows:

```
RUN $LBR.TSK
LBR>BINIT.OLD=[1,1]BP2COM/EX:$BINIT
LBR>^Z                          (CTRL/Z to exit)
```

4. Run the patch utility to merge old module and correction:

```
RUN $PAT.TSK
PAT>BINIT=BINIT.OLD/CS:XXX,BINIT.PAT/CS:4462
PAT>^Z                          (CTRL/Z to exit)
```

where XXX is 7Ø31 for EIS versions
1Ø72Ø for FIS
16340 for FPU

5. Replace patched module in library:

```
RUN $LBR.TSK
LBR>[1,1]BP2COM/RP=BINIT
LBR>^Z                          (CTRL/Z to exit)
```

6. Re-TASK BUILD any programs that used this module.

RSTS/E VØ6C-Ø3 Seq 27.6.6 M
BASIC-PLUS-2
BP2COM Object Library Patches Page 1 of 1

DEBUG MAY IGNORE BREAK IN USER FUNCTION - MANDATORY BP2COM PATCH

PROBLEM:

When using DEBUG, a line which contains a user function call will not BREAK after the function has been executed once.

SOLUTION:

This required patch to the object library file (BP2COM.OLB) will cure the problem.
NOTE: This patch duplicates Article 27.5.1 for this component.

PROCEDURE:

1. Build the following correction file using an editor and name it: DEBUG.COR

```
.TITLE  $DEBUG
.IDENT  /Ø7REA/
.PSECT  $DEBUG
.=.+364
MOV     R4,@RØ
ADD     #2,@RØ
.END
```

2. Assemble the correction file as follows:

```
RUN $MAC.TSK
MAC>DEBUG.PAT,DEBUG.LIS=DEBUG.COR
MAC>^Z                          (CTRL/Z to exit)
```

3. Extract the original module as follows:

```
RUN $LBR.TSK
LBR>DEBUG.OLD=[1,1]BP2COM/EX:$DEBUG
LBR>^Z                          (CTRL/Z to exit)
```

4. Run the patch utility to merge old module and correction:

```
RUN $PAT.TSK
PAT>DEBUG=DEBUG.OLD/CS:166557,DEBUG.PAT/CS:7413
PAT>^Z                          (CTRL/Z to exit)
```

5. Replace patched module in library:

```
RUN $LBR.TSK
LBR>[1,1]BP2COM/RP=DEBUG
LBR>^Z                          (CTRL/Z to exit)
```

6. Re-TASK BUILD any programs that used this module.

RSTS/E VØ6C-Ø3 Seq 27.6.7 M
BASIC-PLUS-2
BP2COM Object Library Patches Page 1 of 2

AN ATTEMPT TO USE A SUBSCRIPT OUT OF RANGE IN DEBUG WILL GIVE MEMORY MANAGEMENT ERRORS

PROBLEM:

While in the DEBUGGER, any attempt to access an array element whose subscript is out of range (or, in fact, any normally trappable error) will generate a MEMORY MANAGEMENT VIOLATION.

SOLUTION:

This required patch to the object library file (BP2COM.OLB) will cure the problem. NOTE: This patch duplicates Article 27.5.2 for this component. Furthermore, the patch in Article 27.6.6 must have been previously installed.

PROCEDURE:

1. Build the following correction file using an editor and name it: DEBUG.COR

```
        .TITLE  $DEBUG
        .IDENT  /Ø7REB/
        .PSECT  $DEBUG
        $$PC=.
        .=$$PC+2514
        MOV     #LPERPT,110(RØ)
        .=$$PC+3144
2$:
        .=$$PC+3256
3$:     MOV     @#52,RØ
        BR      1$
        .=$$PC+3274
        MOV     #LPERPT,110(RØ)
        .=$$PC+3356
        BR      2$
        .=$$PC+3430
        MOV     #2$,@R4
        .=$$PC+3444
1$:     MOV     340(RØ),SP
        CLR     144(RØ)
        JSR     PC,$$PC+1256
        BR      2$
LPERPT: $$PC+3252
        .END
```

2. Assemble the correction file as follows:

```
RUN $MAC.TSK
MAC>DEBUG.PAT,DEBUG.LIS=DEBUG.COR
MAC>^Z                          (CTRL/Z to exit)
```

3. Extract the original module as follows:

```
RUN $LBR.TSK
LBR>DEBUG.OLD=[1,1]BP2COM/EX:$DEBUG
LBR>^Z                          (CTRL/Z to exit)
```

4. Run the patch utility to merge old module and correction:

```
RUN $PAT.TSK
PAT>DEBUG=DEBUG.OLD/CS:171271,DEBUG.PAT/CS:17400
PAT>^Z                          (CTRL/Z to exit)
```

5. Replace patched module in library:

```
RUN $LBR.TSK
LBR>[1,1]BP2COM/RP=DEBUG
LBR>^Z                          (CTRL/Z to exit)
```

6. Re-TASK BUILD any programs that used this module.

RSTS/E VØ6C-Ø3 Seq 27.7.1 M
BASIC-PLUS-2
BASRMS Object Library Patches Page 1 of 1

SCRATCH STATEMENT PRODUCES UNIBUS TIMEOUT ERRORS - MANDATORY BASRMS OLB PATCH

PROBLEM:

Any program attempting to execute a SCRATCH statement on an RMS file receives a UNIBUS TIMEOUT error.

SOLUTION:

This required patch to the object library file (BASRMS.OLB) will cure the problem.

PROCEDURE:

1. Build the following correction file using an editor and name it: ISSCR.COR

```
.TITLE  $ISSCR
.IDENT  /Ø2CMA/
.PSECT
.=.+46
JMP     @(R4)+
.END
```

2. Assemble the correction file as follows:

```
RUN $MAC.TSK
MAC>ISSCR.PAT,ISSCR.LIS=ISSCR.COR
MAC>^Z                          (CTRL/Z to exit)
```

3. Extract the original module as follows:

```
RUN $LBR.TSK
LBR>ISSCR.OLD=[1,1]BASRMS/EX:$ISSCR
LBR>^Z                          (CTRL/Z to exit)
```

4. Run the patch utility to merge old module and correction:

```
RUN $PAT.TSK
PAT>ISSCR=ISSCR.OLD/CS:35223,ISSCR.PAT/CS:3677
PAT>^Z                          (CTRL/Z to exit)
```

5. Replace patched module in library:

```
RUN $LBR.TSK
LBR>[1,1]BASRMS/RP=ISSCR
LBR>^Z                          (CTRL/Z to exit)
```

6. Re-TASK BUILD any programs that used this module.

RSTS/E VØ6C-Ø3
BASIC-PLUS-2
BASRMS Object Library Patches

FILE ACTIVITY AFTER AN INDEXED OPEN MAY PRODUCE CERTAIN RMS ERRORS - MANDATORY BASRMS PATCH

PROBLEM:

If an OPEN or CLOSE of any organization is attempted after an INDEXED OPEN but before the corresponding CLOSE, the program may generate any of several errors of the form: "RMS ERROR -X. This condition is caused by a failure to clear the XAB pointer in the FAB.

SOLUTION:

This required patch to the object library file (BASRMS.OLB) will cure the problem.

PROCEDURE:

1. Build the following correction file using an editor and name it: IXOPN.COR

```
        .TITLE  $IXOPN
        .IDENT  /21CMA/
        .PSECT
        .GLOBL  $BSH03
        .ENABL  AMA
        $$PC=.
        .=$$PC+1Ø1Ø
        JSR     PC,PATCH
        .=$$PC+1124
PATCH:  CLR     34(R5)
        JMP     $BSH03
        .END
```

2. Assemble the correction file as follows:

```
RUN $MAC.TSK
MAC>IXOPN.PAT,IXOPN.LIS=IXOPN.COR
MAC>^Z                          (CTRL/Z to exit)
```

3. Extract the original module as follows:

```
RUN $LBR.TSK
LBR>IXOPN.OLD=[1,1]BASRMS/EX:$IXOPN
LBR>^Z                          (CTRL/Z to exit)
```

4. Run the patch utility to merge old module and correction:

```
RUN $PAT.TSK
PAT>IXOPN=IXOPN.OLD/CS:166755,IXOPN.PAT/CS:7Ø16
PAT>^Z                          (CTRL/Z to exit)
```

5. Replace patched module in library:

```
RUN $LBR.TSK
LBR>[1,1]BASRMS/RP=IXOPN
LBR>^Z                          (CTRL/Z to exit)
```

6. Re-TASK BUILD any programs that used this module.

RSTS/E VØ6C-Ø3
RPG II for RSTS/E VØ6C-Ø3
Package Notes

RPG II V7.Ø6 CAN NOT BE INSTALLED ON RSTS/E VØ6C-Ø3

The RPG II V7.Ø6 kit can not be installed on VØ6C of RSTS/E. An update of RPG II V7 will be released that will run on RSTS/E VØ6C, that will contain all existing corrections, and that will use the RSX Run-Time System instead of RTSLIB. (Also, RPG II V8 will soon be released. This uses RMS-11 and runs on either VØ6B or VØ6C.) Until that time, continue to run RPG II V7.Ø6 on RSTS/E VØ6B.

7.2 Commonly Used System Programs (CUSPs) Status Report

This report reflects the current Version/Edit level of all of the Commonly Used System Programs (CUSPs) supplied as part of the standard RSTS/E V06C-03 System Library Programs release.

System Utilities Package

BUILD V06C-03	DIRECT V06C-03	DISPLY V06C-03	VT05 V06C-03
VT50 V06C-03	DSKINT V06C-03	GRIPE V06C-03	INIT V06C-03
INUSE V06C-03	LOGIN V06C-03	LOGOUT V06C-03	MONEY V06C-03
ODT V06C-03	PLEASE V06C-03	PRIOR V06C-03	QUOLST V06C-03
REACT V06C-03	REORDR V06C-03	SHUTUP V06C-03	SWITCH V06C-03
SYSCAT V06C-03	SYSTAT V06C-03	TALK V06C-03	TTYSET V06C-03
UMOUNT V06C-03	UTILTY V06C-03	UTILT1 V06C-03	

Data Manipulation Package

BPCREF V06C-03	BPCRF1 V06C-03	COPY V06C-03	EDIT V06C-03
EDITCH V06C-03	FILCOM V06C-03	FLINT V06C-03	FLX V06C-03
MAKRTS V06C-03	PIPSML V06C-03	PIPEXT V06C-03	PMDUMP V06C-03
RUNOFF V06C-03			

Error Control Package

ANALY1 V06C-03	ANALYS V06C-03	ERRBLD V06C-03	ERRCPY V06C-03
ERRDET V06C-03	ERRDIS V06C-03	ERRINT V06C-03	

Backup Package

BACCOM V06C-03	BACCON V06C-03	BACDEL V06C-03	BACDIR V06C-03
BACDMP V06C-03	BACDSK V06C-03	BACENT V06C-03	BACFRM V06C-03
BACLAB V06C-03	BACLOD V06C-03	BACLST V06C-03	BACMNT V06C-03
BACPRM V06C-03	BACKTO V06C-03	BACKUP V06C-03	

Spooling and Operator Services Package

OPSER V06C-03	OPSRUN V06C-03		
QUE V06C-03	QUEMAN V06C-03	QUMRUN V06C-03	
CHARS V06C-03	SPOOL V06C-03	SPLIDL V06C-03	SPLRUN V06C-03
BATDCD V06C-03	BATCH V06C-03	BATIDL V06C-03	BATDEC V06C-03
BATRUN V06C-03			

RSTS/E 2780 Package

RJ2780 V06C-03

7.3 Commonly Used System Programs (CUSPs) Protection codes

This table lists the compiled file (.BAC) protection codes for the Commonly Used System Programs (CUSPs) supplied as part of the standard RSTS/E VØ6C-Ø3 System Library Programs release.

Code	Compile sequence
<124>	COMPILE $name
<1Ø4>	COMPILE $name<4Ø>
<232>	COMPILE $name RUN $PIP #=$name.BAC<232>/RE #^Z

System Utilities Package

BUILD <124>	DIRECT <232>	VT5DPY <232>	
VT5ØPY <232>	DSKINT <124>	GRIPE <232>	INIT <124>
INUSE <1Ø4>	LOGIN <232>	LOGOUT <232>	MONEY <1Ø4>
ODT <124>	PLEASE <232>	PRIOR <124>	QUOLST <232>
REACT <124>	REORDR <124>	SHUTUP <124>	SWITCH <232>
SYSCAT <124>	SYSTAT <232>	TALK <232>	TTYSET <232>
UMOUNT <232>	UTILTY <124>	UTILT1 <124>	

Data Manipulation Package

BPCREF <1Ø4>	BPCRF1 <1Ø4>	COPY <1Ø4>	EDIT <1Ø4>
EDITCH <1Ø4>	FILCOM <1Ø4>	FLINT <1Ø4>	FLX <1Ø4>
MAKRTS <124>	PIPSML <1Ø4>	PIPEXT <1Ø4>	PMDUMP <1Ø4>
RUNOFF <1Ø4>			

Error Control Package

ANALY1 <124>	ANALYS <124>	ERRBLD <124>	ERRCPY <124>
ERRDET <124>	ERRDIS <124>	ERRINT <124>	

Backup Package

BACCOM <232>	BACCON <232>	BACDEL <232>	BACDIR <232>
BACDMP <232>	BACDSK <232>	BACENT <124>	BACFRM <232>
BACLAB <232>	BACLOD <232>	BACLST <232>	BACMNT <232>
BACPRM <124>	BACKTO <232>	BACKUP <232>	

Spooling and Operator Services Package

OPSER <124>	OPSRUN <124>		
QUE <232>	QUEMAN <124>	QUMRUN <124>	
CHARS <124>	SPOOL <124>	SPLIDL <124>	SPLRUN <124>
BATDCD <124>	BATCH <124>	BATIDL <124>	BATDEC <124>
BATRUN <124>			

8.Ø Software Product Descriptions (SPDs) and Option Bulletins

8.1 Software Product Descriptions (SPD's)

Periodically, new or revised Software Product Description (SPD) bulletins will appear in the Software Dispatch for various software products marketed by DIGITAL. This section of the notebook is provided as a convenient place to file these SPDs for future reference.

Every SPD has a unique number, a sample of which is shown below:

SPD 14.52.Ø

This number appears on the first page of the SPD bulletin, at the far right end of the PRODUCT NAME: line.

Note that the last part of the SPD number is the revision level, with Ø being assigned for original issues of SPDs.

8.2 Option Bulletins

Periodically, new or revised Option Bulletins will appear in the Software Dispatch for various products marketed by DIGITAL. This section of the release notes is provided as a convenient place to file these bulletins. There is no filing scheme for Option Bulletins.

Appendix A SYSGEN Editor Manual

RSTS/E system generations are normally performed under a RSTS/E SYSGEN monitor configured for all supported disk and tape devices. During the system generation process the RT11 emulator serves as the keyboard monitor and Run-Time System for the various utility programs. In some special cases, it may be necessary to use a text editor during SYSGEN to modify batch or configuration files. Since the BASIC-PLUS Run-Time System is normally not available at this time, the standard RSTS/E EDIT program cannot be used. To allow modification of files during system generation, the RT11 Editor (which runs under the emulator) is included in the RSTS/E distribution kits. RT11 EDIT is supplied only for SYSGEN purposes.

The RT11 EDIT manual is reproduced on the following pages. This information was extracted verbatim from the RT11 System User's Guide; certain sections do not apply in the RSTS/E environment. Specifically, the display editor features presented in Section 5.7 do not apply and should be ignored. Also, to invoke RT11 EDIT under RSTS/E, the EDIT command is not used. Rather, type

```
R EDIT <CR>
```

to enter EDIT, then use either the ER, EW, or EB commands to open files for editing (see Section 5.6).

CHAPTER 5
TEXT EDITOR

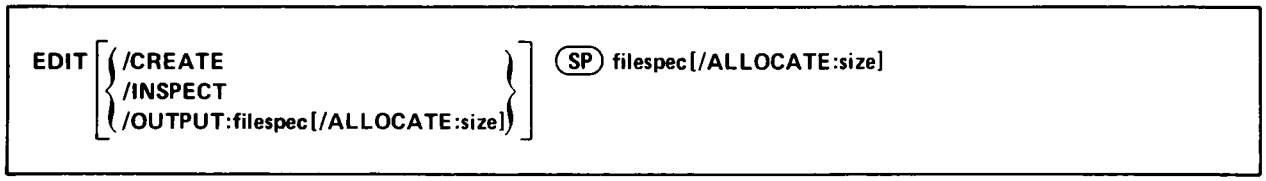
The text editor (EDIT) is a program that creates or modifies ASCII source files for use as input to other system programs such as the MACRO assembler or the FORTRAN compiler. EDIT, which accepts commands you type at the terminal, reads ASCII files from any input device, makes specific changes, and writes on any output device. EDIT allows efficient use of VT11 or VS60 display hardware, if they are part of the system configuration.

The editor considers a file to be divided into logical units called pages. A page of text is generally 50-60 lines long (delimited by form feed characters) and corresponds approximately to a physical page of a program listing. The editor reads one page of text at a time from the input file into its internal buffers where the page becomes available for editing. You can then use editing commands to:

- Locate text to be changed
- Execute and verify the changes
- List an edited page on the console terminal
- Output a page of text to the output file.

5.1 CALLING AND USING EDIT

You can call the text editor when you are at monitor level. The syntax of the command is:



See Section 4.4 for a description of the EDIT command and its options.

5.2 MODES OF OPERATION

Normally, the editor operates in either command mode or text mode. In command mode the editor interprets all input you type on the keyboard as commands to perform some operation. In text mode the editor interprets all typed input as text to replace, insert into, or append to the contents of the text buffer.

Immediately after being loaded into memory and started, the editor is in command mode. EDIT prints an asterisk at the left margin of the console terminal page to indicate that it is ready to accept a command. Terminate all commands by pressing the ESCAPE key twice in succession. Execution of commands proceeds from left to right. Should EDIT encounter an error before it begins execution of a command string, it prints an error message followed by an asterisk at the beginning of a new line, indicating that it is still in command mode and awaiting a legal command. EDIT does not execute the command in error or any succeeding command. You should retype the command correctly.

To enter text mode, type a command that must be followed by a text string. These commands insert, replace, exchange, or otherwise manipulate text. When you type one of these commands, EDIT recognizes all succeeding characters as part of the text string until it encounters an ESCAPE character. The ESCAPE terminates the text string and causes the editor to reenter command mode.

You can use a special editing mode, called immediate mode, whenever the VT-11 display hardware is running. Section 5.7.2 describes this mode.

5.3 SPECIAL KEY COMMANDS

Table 5-1 lists the EDIT key commands. Type a control command by holding down the CTRL key while typing the appropriate character.

Table 5-1 EDIT Key Commands

Key	Explanation
ESCAPE, ALTMODE, or SEL	Echoes $. A single ESCAPE terminates a text string. A double ESCAPE (two consecutive ESCAPEs) executes the command string. For example: `*GMOV A,B$-1D$$` The first ESCAPE ($) terminates the text object (MOV A,B) of the Get command. The double ESCAPE ($$) terminates the Delete command and executes the entire command string. In this example, the character B will be deleted as a result of execution.

CTRL/C	Echoes at the terminal as ^C. If EDIT encounters a CTRL/C as a command in command mode, it terminates execution and returns control to the monitor. You can restart the editor by typing R EDIT or REENTER in response to the monitor's prompt. If EDIT encounters a CTRL/C in a text object, EDIT includes the CTRL/C in the text object, just like any other character. If the editor is executing a lengthy command and you want to stop EDIT, type two CTRL/C commands in succession. This will abort the command, generate the ?EDIT-F-COMMAND ABORTED error message, and return the editor to command mode. For example:
	
	`*I^C^C^C$$`
	`*^C$$`
	
	In the first command, the three CTRL/C characters are part of the text object of the Insert command. EDIT treats them like any other character. In the second command string, the CTRL/C occurs at command level, and causes the editor to terminate.
	
	If no commands (other than CLOSE) are executed between the time you terminate the editor and the time you issue a REENTER command, the text buffer is preserved exactly as it was at program termination. However, only the text buffer is preserved. The input and output files are closed, and the save and macro buffers are reinitialized.
	
	If you inadvertently terminate an editing session before the output file can be closed, you can often use the monitor CLOSE command to make permanent the portion of the output file that has already been written (see Section 4.4). You can then reenter the editor, open a new output file, and continue the editing session.

(Continued on next page)

Table 5-1 (Cont.) EDIT Key Commands

Key	Explanation
CTRL/O	Echoes ^O and a carriage return. Inhibits printing on the terminal until completion of the current command string. Typing a second CTRL/O resumes output.
CTRL/U	Echoes ^U and a carriage return. Deletes all the characters on the current terminal input line. (Equivalent to pressing the RUBOUT key until all the characters back to the beginning of the line are deleted.)
RUBOUT or DELETE	Deletes a character from the current command line; echoes a backslash followed by the character deleted. Each succeeding RUBOUT you type deletes and echoes another character. An enclosing backslash prints when you type a key other than RUBOUT. This erasure is done from right to left. Since EDIT accepts multiple line commands, RUBOUT can delete past the carriage return/line feed combination and delete characters on the previous line. You can use RUBOUT in both command and text modes.
TAB	Spaces to the next tab stop. Tab stops are positioned every eight spaces on the terminal; pressing the TAB key causes the carriage to advance to the next tab position.
CTRL/X	Echoes ^X and a carriage return. CTRL/X causes the editor to ignore the entire command string you are currently entering. The editor prints a carriage return/line feed combination and an asterisk to indicate that you can enter another command. For example: *IABCD EFGH^X * A CTRL/U would cause only deletion of EFGH; CTRL/X erases the entire command.

5.4 COMMAND STRUCTURE

EDIT commands fall into eight general categories. Table 5-2 lists these categories and the commands they include.

Table 5-2 EDIT Command Categories

Category	Commands	Section
File open and close	Edit Backup	5.6.1.3
	Edit Read	5.6.1.1
	Edit Write	5.6.1.2
	End File	5.6.1.4
File input/output	EXit	5.6.2.4
	Next	5.6.2.3
	Read	5.6.2.1
	Write	5.6.2.2

(Continued on next page)

Table 5-2 (Cont.) EDIT Command Categories

Category	Commands	Section
Immediate mode	ESCAPE	5.7.2
	CTRL D	5.7.2
	CTRL G	5.7.2
	CTRL N	5.7.2
	CTRL V	5.7.2
	RUBOUT	5.7.2
Pointer location	Advance	5.6.3.3
	Beginning	5.6.3.1
	Jump	5.6.3.2
Search	Find	5.6.4.2
	Get	5.6.4.1
	Position	5.6.4.3
Text listing	List	5.6.5.1
	Verify	5.6.5.2
Text modification	Change	5.6.6.4
	Delete	5.6.6.2
	eXchange	5.6.6.5
	Insert	5.6.6.1
	Kill	5.6.6.3
Utility	Edit Console	5.7.1
	Edit Display	5.7.1
	Edit Lower	5.6.7.6
	Edit Upper	5.6.7.6
	Edit Version	5.6.7.5
	Executive Macro	5.6.7.4
	Macro	5.6.7.3
	Save	5.6.7.1
	Unsave	5.6.7.2

The general syntax for all the EDIT commands, with the exception of the immediate mode commands, is:

[n]C[text]$

or

[n]C$

where

n represents one of the legal arguments from Table 5-3.

C represents a 1- or 2-letter command.

text represents a string of successive ASCII characters.

As a rule, commands are separated from one another by a single ESCAPE; however, if the command requires no text, the separating ESCAPE is not necessary. Commands are terminated by a single ESCAPE; typing a second ESCAPE begins execution. (You use ESCAPE differently when immediate mode is in effect; Section 5.7.2 details its use in this case.)

The syntax of display editor commands is somewhat different from the normal editing command format, and is described in Section 5.7.

5.4.1 Arguments

An argument is positioned before a command letter. It specifies either the particular portion of text to be affected by the command or the number of times to perform the command. With some commands, this specification is implicit and no argument is needed; other editing commands require an argument. Table 5-3 lists the possible arguments and their meanings.

Table 5-3 Command Arguments

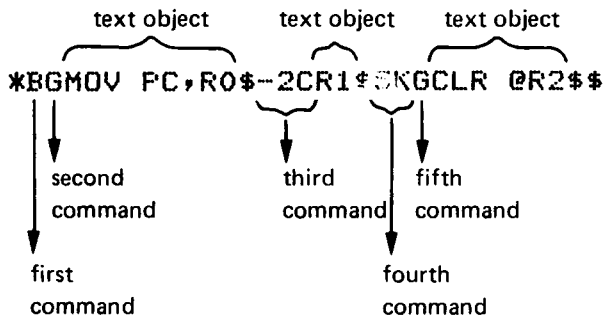
Argument	Meaning
n	Stands for any integer in the range -16383 to +16383 and may, except where noted, be preceded by a plus (+) or minus (-) sign. If no sign precedes n, it is assumed to be a positive number. The absence of n implies a 1 (or -1 if a minus sign precedes a command). n can represent the number of characters or lines forward or backward (+ or -) to move the pointer, or it can represent the number of times to execute the operation.
0	Indicates the text between the beginning of the current line and the reference pointer (see Section 5.4.3).
/	Refers to the text between the reference pointer and the end of the text in the buffer.
=	Use only with the J, D, and C commands to represent -n, where n is equal to the length of the last text argument used.

The roles of all arguments are explained more specifically in the following sections.

5.4.2 Command Strings

All EDIT command strings are terminated by two successive ESCAPE characters. Use spaces, carriage returns, and line feeds within a command string to increase command readability. EDIT ignores them unless they appear in a text string. Commands to insert text can contain text strings that are several lines long. Each line you enter is terminated by the carriage return key, which inserts both a carriage return and a line feed character into the text. The entire command is terminated by a double ESCAPE.

You can string several commands together and execute them in sequence. For example:



where

B	is the first command.
GMOV PC,R0	is the second command (MOV PC,R0 is the text object).
-2CR1	is the third command (R1 is the text object).
5K	is the fourth command.
GCLR @R2	is the fifth command (CLR @R2 is the text object).
$	separates the end of each text object from the following command.
$$	executes the commands.

Execution of a command string begins when you type the double ESCAPE and proceeds from left to right. Except when they are part of a text string, EDIT ignores spaces, carriage returns, line feeds, and single ESCAPEs. For example:

```
*BGMOV R0$=CCLR R1$AV$$
```

You can also type this command as:

```
*B$ GMOV R0$
=CCLR R1$
A$ V$$
```

Execution of the two commands will be the same.

5.4.3 The Current Location Pointer

Most EDIT commands function with respect to a movable reference pointer that is normally located between the most recent character operated upon and the next character in the buffer. It is important to think of this pointer as being between two characters and never directly on a character. At the start of editing operations, the pointer precedes the first character in the buffer, although it is not displayed on the console terminal. At any given time during the editing procedure, think of the pointer as representing the current position of the editor in the text. The pointer moves during editing operations according to the type of editing operation being performed. Refer to text in the buffer as so many characters or lines preceding or following the pointer.

5.4.4 Character- and Line-Oriented Command Properties

Edit commands are either character-oriented or line-oriented: character-oriented commands affect a specified number of characters preceding or following the pointer; line-oriented commands operate on entire lines of text.

The argument of character-oriented commands specifies the number of characters in the buffer on which to operate. If n is unsigned (positive), the command operates in a forward direction. If n is preceded by a minus sign (negative), the command moves the reference pointer backwards. (LF), (RET), and null characters, although not printed. are embedded in text lines, counted as characters in character-oriented commands, and treated as any other text characters. When you press the (RET) key, both a carriage return and a line feed character are inserted into the text. For example, assume the pointer is positioned as indicated in the following text (↑ represents the current position of the pointer):

```
MOV   #VECT,R2(RET)(LF)↑
CLR   @R2(RET)(LF)
```

The EDIT command -2J moves the pointer back two characters to precede the carriage return character.

```
MOV    #VECT, R2↑(RET)(LF)
CLR    @R2 (RET)(LF)
```

The command 10J advances the pointer forward by ten characters and places it between the (RET) and (LF) characters at the end of the second line. Note that the tab character preceding @R2 is also counted as a single character.

```
MOV    #VECT,R2 (RET)(LF)
CLR    @R2 (RET)↑(LF)
```

Finally, to place the pointer after the C in the first line, use a -14J command. The J (Jump) command is explained in Section 5.6.3.2.

```
MOV    #VECT↑,R2 (RET)(LF)
CLR    @R2 (RET)(LF)
```

When you use line-oriented commands, the argument of the commands specifies the number of lines on which to operate. Because EDIT counts the line-terminating characters to determine the number of lines on which to operate, an argument, n, does not affect the same number of lines forward (positive) as it affects backward (negative). For example, the argument -1 applies to the line beginning with the first character following the second previous end-of-line and ending with the character preceding the pointer. The argument 1 in a line-oriented command, however, applies to the text beginning with the first character following the pointer and ending at the first end-of-line. Thus, if the pointer is at the center of the line, the argument -1 affects one and one-half lines backwards from the pointer and the argument 1 affects one-half line beyond the pointer.

For example, assume the buffer contains:

```
MOV    ↑PC,R1 (RET)(LF)
ADD    #DRIV-.,R1(RET)(LF)
MOV    #VECT,R2 (RET)(LF)
CLR    @R2(RET)(LF)
```

The command to advance the pointer one line (1A) causes the following change:

```
MOV    PC,R1(RET)(LF)
↑ADD   #DRIV-.,R1(RET)(LF)
MOV    #VECT,R2 (RET)(LF)
CLR    @R2(RET)(LF)
```

The command 2A moves the pointer over two (RET)(LF) combinations to precede the fourth line:

```
MOV    PC,R1 (RET)(LF)
ADD    #DRIV-.,R1(RET)(LF)
MOV    #VECT,R2 (RET)(LF)
↑CLR   @R2 (RET)(LF)
```

Assume the buffer contains:

```
MOV    PC,R1 (RET)(LF)
ADD    #DRIV-.,R1(RET)(LF)
MOV    #VECT,R2 (RET)(LF)
CLR    @R2 ↑(RET)(LF)
```

A command of -1A moves the pointer back by one and one-half lines to precede the second line.

```
MOV   PC,R1(RET)(LF)
↑ADD  #DRIV-.,R1(RET)(LF)
MOV   #VECT,R2(RET)(LF)
CLR   @R2(RET)(LF)
```

Now a command of -1A moves the pointer back by only one line.

```
↑MOV  PC,R1(RET)(LF)
ADD   #DRIV-.,R1(RET)(LF)
MOV   #VECT,R2(RET)(LF)
CLR   @R2(RET)(LF)
```

5.4.5 Command Repetition

You can execute portions of a command string more than once by enclosing the portion in angle brackets (<>) and preceding the left angle bracket with the number of iterations you desire. The syntax is:

```
n<command>
```

For example:

```
C1$C2$n<C3$C4$>C5$$
```

where

C represents a command.

n represents an iteration argument.

Commands C1 and C2 each execute once, then commands C3 and C4 execute n times. Finally, command C5 executes once and the command line is finished. The iteration argument (n) must be a positive number (in the range 1 through 16,383) and, if you do not specify it, it is assumed to be 1. If the number is negative or too large, an error message prints. You can nest iteration brackets up to 20 levels. EDIT checks command lines to make certain the brackets are correctly used and match prior to execution.

Essentially, enclosing a portion of a command string in iteration brackets and preceding it with an iteration argument (n) is equivalent to typing that portion of the string n times. For example:

```
*BGAAA$3<-DIB$-J>V$$
*BGAAA$-DIB$-J-DIB$-J-DIB$-JV$$
```

These two strings are equivalent.

Similarly, the following two strings are equivalent:

```
*B3<2<AD>V>$$
*BADADVADADVADADV$$
```

The following bracket structures are examples of legal usage:

```
<<><<<><>>>>
<<<>>><><>
```

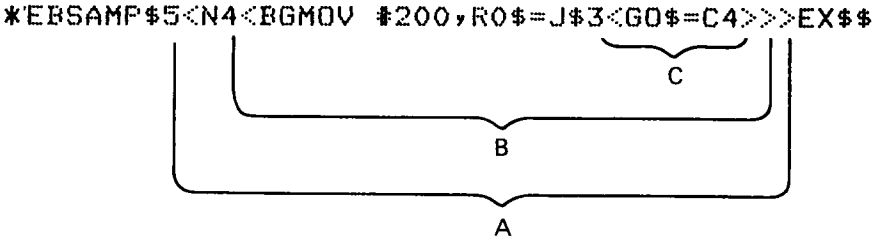
The following bracket structures are examples of illegal combinations that will cause an error message since the brackets are not properly matched:

```
><><
<<<>>
```

During command repetition, execution proceeds from left to right until a right bracket is encountered. EDIT then returns to the last left bracket encountered, decreases the iteration counter, and executes the commands within the brackets. When the counter is decreased to 0, EDIT looks for the next iteration count to the left and repeats the same procedures. The overall effect is that EDIT works its way to the innermost brackets and then works its way back again. The most common use for iteration brackets is found in commands, such as Unsave (U), that do not accept repeat counts. For example:

```
*3<U>$$
```

Assume you want to read a file called SAMP (stored on device DK:), and you want to change the first four occurrences of the instruction MOV #200,R0 on each of the first five pages to MOV #244,R4. Enter the following command line:

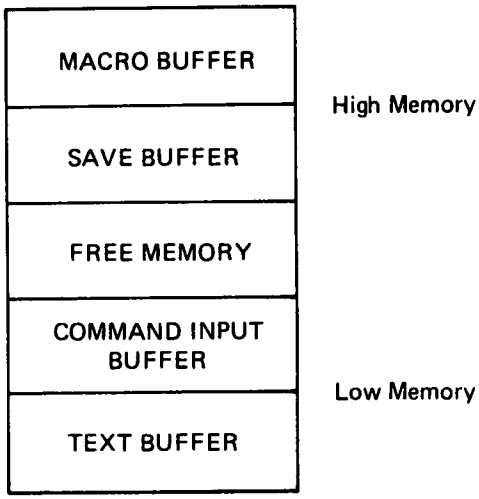


The command line contains three sets of iteration loops (A,B,C) and executes as follows:

Execution initially proceeds from left to right; EDIT opens the file SAMP for input and reads the first page into memory. EDIT moves the pointer to the beginning of the buffer and initiates a search for the character string MOV #200,R0. When it finds the string, EDIT positions the pointer at the end of the string, but the =J command moves the pointer back, so that it is positioned immediately preceding the string. At this point, execution has passed through each of the first two sets of iteration loops (A,B) once. The innermost loop (C) is next executed three times, changing the 0s to 4s. Control now moves back to pick up the second iteration of loop B, and again moves from left to right. When loop C has executed three times, control again moves back to loop B. When loop B has executed a total of four times, control moves back to the second iteration of loop A, and so forth, until all iterations have been satisfied.

5.5 MEMORY USAGE

The memory area used by the editor is divided into four logical buffers as follows:



The text buffer contains the current page of text you are editing, and the command input buffer holds the command you are currently typing at the terminal. If a command you are currently entering is within ten characters of exceeding the space available in the command buffer, the following message prints on the terminal.

```
?EDIT-W-Command buffer almost full
```

If you can complete the command within ten characters, you can finish entering the command; otherwise you should press the ESCAPE key twice to execute that portion of the command line already completed. The message prints each time you enter a character in one of the last ten spaces.

If you attempt to enter more than ten characters, EDIT prints the following message and aborts the command.

```
?EDIT-F-Command buffer full;no command(s) executed
```

This will never occur if you heed the preceding warning and terminate the command immediately.

The save buffer contains text stored with the Save (S) command, and the macro buffer contains the command string macro entered with the Macro (M) command. (Both commands are explained in Section 5.6.7.)

EDIT does not allocate space for the macro and save buffers until an M or S command executes. Once you enter an M or S command, a 0M or 0U (Unsave) command returns that space to the free area.

The size of each buffer automatically expands and contracts to accommodate the text you are entering; if there is not enough space available to accommodate required expansion of any of the buffers, EDIT prints the error message:

```
?EDIT-F-Insufficient memory
```

5.6 EDITING COMMANDS

This section describes the commands and procedures required to:

- Read text from the input files to the buffer
- Create a backup version of the file
- List the contents of the buffer on the terminal
- Move the reference pointer

- Locate specific characters or strings of characters within the text buffer
- Insert, relocate, or delete text in the buffer
- Close the output file
- Terminate the editing session.

The following sections are arranged, in order, by category of command function, as illustrated in Table 5-2.

5.6.1 File Open and Close Commands

You can use file open and close commands to:

- Open an existing file for input and prepare it for editing
- Open a file for output of newly created or edited text
- Open an existing file for editing and create a backup version of it
- Close an open output file.

5.6.1.1 Edit Read – The Edit Read (ER) command opens an existing file for input and prepares it for editing. Only one file can be open for input at a time.

The syntax of the command is:

```
ERdev:filnam.typ$
```

The string argument (dev:filnam.typ) is limited to 19 characters and specifies the file to be opened. If you do not specify a device, DK: is assumed. If a file is currently open for input, EDIT closes the file and opens the new one.

Edit Read does not input a page of text nor does it affect the contents of the other user buffers.

You can use Edit Read on a file that is already open to close that file for input and reposition EDIT at its beginning. The first Read command following any Edit Read command inputs the first page of the file.

```
*ERDT1:SAMP.MAC$$
```

This command string, for example, opens the file SAMP.MAC on device DT1: for input.

NOTE

If you enter EDIT with the monitor EDIT/INSPECT or EDIT/OUTPUT command, an Edit Read command is automatically performed on the file named in the EDIT command.

5.6.1.2 Edit Write – The Edit Write (EW) command opens a file for output of newly created or edited text. However, no text is output and the contents of the buffers are not affected. Only one file can be open for output at a time. EDIT closes any output files currently open and preserves any edits made to the file.

The syntax of the command is:

```
EWdev:filnam.typ[n]$
```

The string argument (dev:filnam.typ[n]) is limited to 19 characters and is the name you assign to the output file being opened. If you do not specify a device, DK: is assumed. [n] is an optional decimal number that represents the length of the file to be opened. Note that the square brackets [] are part of the argument, n. You must type them. If you do not specify [n], the default size will be used. That is, the system will choose the larger of 1) one-half the largest available space, and 2) the second largest available space. If this is not adequate for the output file size, you must close this file and open another when this one becomes full. You should use the [n] construction whenever there is doubt as to whether enough space is available on the device for one output file.

If a file with the same name already exists on the device, EDIT deletes the existing file when you type an Exit, End File, or another Edit Write command. EDIT prints the warning message:

```
?EDIT-W-Superseding existing file
```

The following command, for example, opens for output the file FILE.BAS on device DK: and allocates 11 blocks of space for it.

```
*EWFILE.BAS[11]$$
```

NOTE

If you enter EDIT with the monitor EDIT/CREATE command, an Edit Write command is automatically performed on the file named in the EDIT command. If you enter EDIT with the monitor EDIT/OUTPUT command, an Edit Write is automatically performed on the file named with the /OUTPUT option.

5.6.1.3 Edit Backup – Use the Edit Backup (EB) command to open an existing file for editing and at the same time create a backup version of the file. EDIT closes any input and output file currently opened. No text is read or written with this command.

The syntax of the command is:

```
EBdev:filnam.typ[n]$
```

The device designation, file name, and file type are limited to 19 characters. If you do not specify a device, DK: is assumed. [n] is optional and represents the length of the file to be opened; if you do not specify [n], the default size will be used. That is, the system will choose the larger of 1) one-half the largest available space, and 2) the second largest available space.

The file you indicate in the command line must already exist on the device you designate, since text will be read from this file as input. At the same time, EDIT opens an output file under the same file name and file type. When the output file is closed, EDIT renames the original file (used as input) with the current file name and a .BAK file type and deletes any previous file with this file name and a .BAK file type. EDIT closes the new output file and assigns it the name you specify in the EB command. This renaming of files takes place when an Exit, End File, or subsequent Edit Write or Edit Backup command executes. If you terminate the editing session with a CTRL/C command before the output file is closed, the new output file is not made permanent, and the renaming of the current version to .BAK does not take place.

```
*EBSY:BAS1.MAC$$
```

This command opens BAS1.MAC on device SY:. When editing is complete, the old BAS1.MAC becomes BAS1.BAK, and the new file becomes BAS1.MAC. EDIT deletes any previous version of BAS1.BAK.

NOTE

In EB, ER, and EW commands, leading spaces between the command and the file name are not permitted because EDIT assumes the file name to be a text string. All dev:filnam.typ specifications for EB, ER, and EW commands conform to the RT-11 conventions for file naming and are identical to file names entered in command strings used with other system programs.

If you enter EDIT with an unqualified monitor EDIT command, an Edit Backup command is automatically performed on the file named in the EDIT command.

5.6.1.4 End File – The End File (EF) command closes the current output file and makes it permanent. You can use the EF command to create an output file from a section of a large input file or to close an output file that is full before you open another file. Modifiers are illegal with an EF command. Note that an implied EF command is included in EW and EB commands.

The syntax of the command is:

EF

Table 5-4 illustrates the relationship between the file open and close commands and the buffers and files themselves.

Table 5-4 EDIT Commands and File Status

Command	Input File	Text Buffer	Output File
ERXXX$	Opens XXX for input; closes existing input file, if any	Unchanged	Unchanged
EWXXX$	Unchanged	Unchanged	Opens XXX for output; closes existing output file, if any; performs .BAK renaming if EB is in effect
EBXXX$	Opens XXX for input; closes existing input file, if any	Unchanged	Opens a temporary file for output; closes existing output file, if any; performs .BAK renaming if EB is in effect
EF$	Unchanged	Unchanged	Closes output file; performs .BAK renaming if EB is in effect
EX$	Copies to output file	Copies to output file	Closes output file after copying complete; performs .BAK renaming if EB is in effect

5.6.2 File Input/Output Commands

You use file input/output commands to:

- Read text from an input file into the buffer
- Copy lines of text from the buffer into an output file
- Terminate the editing session.

5.6.2.1 Read – Before you can edit text, you must read the input file into the buffer. The Read (R) command reads a page of text from the input file (previously specified in an ER or EB command) and appends it to the current contents, if any, of the text buffer.

The command is:

R

No arguments are used with the R command. If text resides in the buffer prior to the R command, the pointer does not move; however, if no text resides in the buffer, the pointer is placed at the beginning of the buffer. EDIT transfers text to the buffer until one of the following conditions occurs:

1. A form feed character, signifying the end of the page, is encountered.
2. The text buffer is 500 characters from being full. (When this condition occurs, the Read command inputs up to the next carriage return/line feed combination, then returns to command mode. An asterisk prints as though the read were complete, but text will not have been fully input).
3. An end-of-file is encountered, (the ?EDIT-F-END OF INPUT FILE message prints when all text in the file has been read into memory and no more input is available).

The maximum number of characters that you can bring into memory with an R command depends on the system configuration and the memory requirements of other system components. EDIT prints an error message if the read exceeds the memory available or if no input is available.

The following example edits a file using the EB and R commands.

```
*EBSJK1.BAS$$
```

This command opens SJK1.BAS on DK: and permits modification.

```
*R/L$$
THIS IS PAGE ONE OF
FILE SJK1.BAS.
```

This command reads the first page of SJK1.BAS into the buffer. The pointer is placed at the beginning of the buffer. /L lists the contents of the buffer on the terminal beginning at the pointer and ending with the last character in the buffer.

5.6.2.2 Write – The Write (nW) command copies lines of text from the text buffer to the output file (as specified in the EW or EB command). The contents of the buffer are not altered and the pointer is left unchanged (unless an output error occurs).

NOTE

EDIT uses a system of intermediate buffers to store output before it actually writes the data to an output file. The Write command logically writes to the file, but actual output to a

device does not occur until the intermediate buffer fills. When the editor closes a file (that is, after you issue an EF, EB, EX, or EW command), the editor writes from the buffer to the file and the file is complete. If the editor does not close a file (if you exit with CTRL/C and use the CLOSE command), it is possible that the output file will be missing the last 512 characters.

The syntax of the command is:

nW

The argument you supply with the W command determines the lines of text to copy. Table 5-5 lists the arguments for the W command and their effect.

Table 5-5 Write Command Arguments

Argument	Meaning
n	Writes n lines of text beginning at the pointer and ending with the nth end-of-line character to the output file.
-n	Writes n lines of text to the output file beginning with the first character on the -nth line and terminating at the pointer.
0	Writes to the output file the current line up to the pointer.
/	Writes to the output file the text between the pointer and the end of the buffer.

If the buffer is empty when the write executes, no characters are output.

The following examples illustrate the use of the W command.

```
*5W$$
```

This command writes the five lines of text following the pointer into the current output file.

```
*-2W$$
```

This command writes the two lines of text preceding the pointer into the current output file.

```
*B/W$$
```

This command writes the entire text buffer to the current output file.

NOTE

If an output file fills while a Write command is executing, EDIT prints the ?EDIT-F-OUTPUT FILE FULL message. In this case, EDIT positions the reference pointer after the last character it wrote successfully. You can then use the following recovery procedure:

1. Close the current output file. (EF command)
2. Open a new output file. (EW command)
3. Delete the characters just written by using -nD or -nK, where n is any arbitrary number that exceeds the number of lines or characters in the buffer.
4. Resume output.

5.6.2.3 Next – The Next (nN) command writes the contents of the text buffer to the output file, deletes the text from the buffer, and reads the next page of the input file into the buffer. The pointer is positioned at the beginning of the buffer. The syntax of the command is:

```
nN
```

If you specify the argument n with the Next command, the sequence is executed n times.

If EDIT encounters the end of the input file when trying to execute an N command, it prints ?EDIT-F-END OF INPUT FILE to indicate that no further text remains in the input file. Since the contents of the buffer has already been transferred to the output file, the buffer is empty.

Using the N command is a quick way to write edited text to the output file and set up the next page of text in the buffer. The N command functions as though it were a combination of the Write, Delete, Read, and Beginning commands. (Delete is a text modification command, described in Section 5.6.6.2; the Beginning command is a pointer relocation command, described in Section 5.6.3.1.) Using the N command with an argument is a convenient way to set up text in the buffer, if you already know its page location. The N command operates in a forward direction only; therefore, you cannot specify negative arguments with an N command.

In the following example, an N command copies an input file with more than one page of text to the output file.

```
*EBDK:TEST.MAC$$
```

This command opens the file TEST.MAC on device DK: and creates a new file entitled TEST.MAC for output.

```
*N/L$$
THIS IS PAGE ONE OF
FILE TEST.MAC.
```

This command reads the first page of the input file, TEST.MAC, into the buffer and lists the entire page on the terminal.

```
*N/L$$
?EDIT-F-End of input file
*
```

This command transfers the contents of the buffer to the output file, clears the buffer, and encounters the end of the file. Because it cannot complete the N sequence, EDIT prints ?EDIT-F-END OF INPUT FILE on the terminal. The buffer is empty and the entire input file has been written to the output file.

5.6.2.4 EXit – Type the Exit (EX) command to terminate an editing session. The Exit command does the following:

- Writes the text buffer to the output file
- Transfers the remainder of the input file to the output file
- Closes all open files
- Renames the backup file with a .BAK file type if an EB command is in effect
- Returns control to the monitor.

The command is:

EX

No arguments are accepted. Essentially, Exit copies the remainder of the input file into the output file and returns to the monitor. Exit is legal only when there is an output file open. If an output file is not open and you want to terminate the editing session, return to the monitor with CTRL/C.

NOTE

You must issue an EF or EX command in order to make an output file permanent. If you use CTRL/C to return to the monitor without issuing an EF command, the current output file will not be saved. (You can, however, make permanent that portion of the text file that has already been written out by using the monitor CLOSE command.)

An example of the contrasting uses of the EF and EX commands follows. Assume an input file, SAMPLE, contains several pages of text. The first and second pages of the file will be made into separate files called SAM1 and SAM2, respectively; the remaining pages of text will then make up the file SAMPLE. This can be done using these commands:

```
*EWSAM1$$
*ERSAMPLE$$
*RNEF$$
*EWSAM2$$
*NEF$$
*EWSAMPLE$EX$$
```

Note that the EF commands are not strictly necessary in this example since the EW command closes a currently open output file before opening another.

5.6.3 Pointer Relocation Commands

Pointer relocation commands allow you to change the current location of the reference pointer within the text buffer.

5.6.3.1 Beginning – The Beginning (B) command moves the current location of the pointer to the beginning of the text buffer.

The command is:

B

There are no arguments.

For example, assume the buffer contains:

```
MOVB  5(R1),@R2
ADD   R1,(R2)+
CLR   @R2
MOVB  6(R1),@R2
```

The B command moves the pointer to the beginning of the text buffer.

```
*B$$
```

The text buffer now looks like this:

```
↑MOVB   5(R1),@R2
 ADD    R1,(R2)+
 CLR    @R2
 MOVB   6(R1),@R2
```

5.6.3.2 Jump – The Jump (nJ) command moves the pointer past the specified number of characters in the text buffer. The syntax of the command is:

nJ

Table 5-6 shows the arguments for the J command and their meanings.

Table 5-6 Jump Command Arguments

Argument	Meaning
(+ or -) n	Moves the pointer (forward or backward) n characters.
0	Moves the pointer to the beginning of the current line (equivalent to 0A).
/	Moves the pointer to the end of the text buffer (equivalent to /A).
=	Moves the pointer backward n characters, where n equals the length of the last text argument used.

Negative arguments move the pointer toward the beginning of the buffer; positive arguments move it toward the end. Jump treats carriage returns, line feeds, and form feed characters the same as any other character, counting one buffer position for each one.

The following commands illustrate the use of the J command.

```
*3J$$
```

This command moves the pointer ahead three characters.

```
*-4J$$
```

This command moves the pointer back four characters.

```
*B$$GABC$$=J$$
```

This command moves the pointer so that it immediately precedes the first occurrence of ABC in the buffer.

5.6.3.3 Advance – The Advance (nA) command is similar to the Jump command except that it moves the pointer a specific number of lines (rather than single characters) and leaves it positioned at the beginning of the line. The syntax of the command is:

nA

Table 5-7 lists the arguments for the A command and their meanings.

Table 5-7 Advance Command Arguments

Argument	Meaning
n	Moves the pointer forward n lines and positions it at the beginning of the nth line.
-n	Moves the pointer backward past n carriage return/line feed combinations and positions it at the beginning of the -nth line.
0	Moves the pointer to the beginning of the current line (equivalent to 0J).
/	Moves the pointer to the end of the text buffer (equivalent to /J).

Following are examples that use the A command.

```
*3A$$
```

This command moves the pointer ahead three lines.

Assume the buffer contains:

```
CLR     @R2
```

The following command moves the pointer to the beginning of the current line:

```
*0A$$
```

Now the buffer looks like this:

```
CLR     @R2
```

5.6.4 Search Commands

Use search commands to locate specific characters or strings of characters within the text buffer.

NOTE

Search commands always have positive arguments. They search ahead in the file. This means that you cannot search for a character string that has already been written to the output file. To do this, you must first close the currently open files (with EX) then edit the file that was just used for output (with EB).

5.6.4.1 Get – The Get (nG) command is the basic search command in EDIT. It searches the current text buffer for the nth occurrence of a specific text string starting at the current location of the pointer. If you do not supply the argument n, EDIT searches for the first occurrence of the text object. The search terminates when EDIT either finds the nth occurrence or encounters the end of the buffer. If the search is successful, EDIT positions the pointer to follow the last character of the text object. EDIT notifies you of an unsuccessful search by printing ?EDIT-F-SEARCH FAILED. In this instance, EDIT positions the pointer after the last character in the buffer.

The syntax of the command is:

```
nGtext$
```

The argument (n) must be positive. If you omit it, EDIT assumes it to be 1.

The text string can be any length and must immediately follow the G command. EDIT makes the search on the portion of the text between the pointer and the end of the buffer.

For example, assume the pointer is at the beginning of the buffer shown below.

```
↑MOV    PC,R1
 ADD    #DRIV-.,R1
 MOV    #VECT,R2
 CLR    @R2
 MOVB   5(R1),@R2
 ADD    R1,(R2)+
 CLR    @R2
 MOVB   6(R1),@R2
```

The following command searches for the first occurrence of the characters ADD following the pointer and places the pointer after the searched characters.

```
*GADD$$
```

Now the buffer looks like this:

```
MOV    PC,R1
ADD↑   #DRIV-.,R1
```

The next command searches for the third occurrence of the characters @R2 following the pointer and leaves the pointer immediately following the text object.

```
*3G@R2$$
```

The buffer is changed to:

```
ADD    R1,(R2)+
CLR    @R2↑
```

After successfully completing a search command, EDIT positions the pointer immediately following the text object. Using a search command in combination with =J places the pointer in front of the text object, as follows:

```
*GTEST$=.J$$
```

This command combination places the pointer before TEST in the text buffer.

5.6.4.2 Find – The Find (nF) command starts at the current pointer location and searches the entire input file for the nth occurrence of the text string. If EDIT does not find the nth occurrence of the text string in the current buffer, it automatically performs a Next command and continues the search on the new text in the buffer. When the search is successful, EDIT leaves the pointer immediately following the nth occurrence of the text string. If the search fails (i.e., EDIT detects the end-of-file for the input file and does not find the nth occurrence of the text string), EDIT prints ?EDIT-F-SEARCH FAILED. In this instance, EDIT positions the pointer at the beginning of an empty text buffer. When you use the F command, EDIT deletes the contents of the buffer after writing it to the output file.

The syntax of the command is:

```
nFtext$
```

The argument (n) must be positive. EDIT assumes it to be 1 if you do not supply another value.

You can use an F command to copy all remaining text from the input file to the output file by specifying a non-existent text object. The Find command functions like a combination of the Get and Next commands.

The following example uses the F command.

```
*2FMOVB     6(R1),@R2$$
```

This command searches the entire input file for the second occurrence of the text string MOVB 6(R1),@R2. EDIT places the pointer following the text string. EDIT writes the contents of each unsuccessfully searched buffer to the output file.

5.6.4.3 Position – The Position (nP) command is identical to the find (F) command with one exception. The F command transfers the contents of the text buffer to the output file as each page is unsuccessfully searched, but the P command deletes the contents of the buffer after it is searched, without writing any text to the output file.

The syntax of the command is:

```
nPtext$
```

The argument (n) must be positive. If you omit it, EDIT assumes it to be 1.

The nP command searches each page of the input file for the nth occurrence of the text object starting at the pointer and ending with the last character in the buffer. If EDIT finds the nth occurrence, it positions the pointer following the text object, deletes all pages preceding the one containing the text object, and positions the page containing the text object in the buffer.

If the search is unsuccessful, EDIT clears the buffer and does not transfer any text to the output file. EDIT positions the pointer at the beginning of an empty text buffer.

The position command is a combination of the Get, Delete, and Read commands; it is most useful as a means of placing the pointer in the input file. For example, if your aim in the editing session is to create a new file from the second half of the input file, a position search saves time.

The following example uses the P command.

```
*P3$$
```

This command searches the input file for the first occurrence of the text object, 3. EDIT positions the pointer after the text object.

```
*0L$$
INPUT FILE PAGE 3
```

The command lists on the terminal the current line up to the pointer.

5.6.5 Text Listing Commands

5.6.5.1 List – The List (nL) command prints at the terminal lines of text as they appear in the buffer. The syntax of the command is:

```
nL
```

An argument preceding the L command indicates the portion of text to print. For example, the command, 2L, prints on the terminal the text beginning at the pointer and ending with the second end-of-line character. The pointer is not altered by the L command. Table 5-8 lists arguments and their effect upon the list command.

Table 5-8 List Command Arguments

Argument	Meaning
n	Prints at the terminal n lines beginning at the pointer and ending with the nth end-of-line character.
-n	Prints all characters beginning with the first character on the -nth line and terminating at the pointer.
0	Prints the current line up to the pointer. Use this command to locate the pointer within a line.
/	Prints the text between the pointer and the end of the buffer.

These examples illustrate the use of the L command.

```
*-2L$$
```

This command prints all characters starting at the second preceding line and ending at the pointer.

```
*4L$$
```

This line prints all characters beginning at the pointer and terminating at the 4th carriage return/line feed combination.

Assuming the pointer location is:

```
MOVB  5(R1),@R2
ADD↑  R1,(R2)+
```

The following command prints the previous one and one-half lines up to the pointer:

```
*-1L$$
```

The terminal output looks like this:

```
MOVB   5(R1),@R2
ADD
```

5.6.5.2 Verify – The Verify (V) command prints at the terminal the entire line in which the pointer is located. It provides a ready means of determining the location of the pointer after a search completes and before you give any editing commands. (The V command combines the two commands 0LL.) You can also type the V command after an editing command to allow proofreading of the results. No arguments are allowed with the V command. The location of the pointer does not change.

5.6.6 Text Modification Commands

You can use the following commands to insert, relocate, and delete text in the text buffer.

5.6.6.1 Insert – The Insert (I) command is the basic command for inserting text. EDIT inserts the text you supply at the location of the pointer, then places the pointer after the last character of the new text.

The syntax of the command is:

Itext$

No arguments are allowed with the insert command, and the text string is limited only by the size of the text buffer and the space available. All characters except ESCAPE are legal in the text string. ESCAPE terminates the text string.

NOTE

If you forget to type the I command, the text will be executed as commands.

EDIT automatically protects against overflowing the text buffer during an insert. If the I command is the first command in a multiple command line, EDIT ensures that there will be enough space for the insert to be executed at least once. If repetition of the command exceeds the available memory, an error message prints.

The following example illustrates the use of the I command.

```
*IMOV           #BUFF,R2
MOV             #LINE,R1
MOVB            -1(R2),R0$$
*
```

This command inserts the text at the current location of the pointer and leaves the pointer positioned after R0.

DIGITAL recommends that you insert large amounts of text into the file in small sections rather than all at once. This way, you are less vulnerable to loss of time and effort due to machine failure or human error. This is the recommended technique for inserting large amounts of text:

1. Open the file with the EB command
2. Insert or edit a few pages of text
3. Insert a unique text string (like ????) to mark your place
4. Use the Exit command to preserve the work you have done so far
5. Start again, using the F command to search for the unique string you used to mark your place
6. Delete your marker and continue editing.

By using this procedure, you reduce your loss (should there be a machine or human error) to the few pages of text on which you just worked.

5.6.6.2 Delete – The Delete (nD) command is a character-oriented command that deletes n characters in the text buffer beginning at the current location of the pointer. The syntax of the command is:

nD

If you do not specify n, EDIT deletes the character immediately following the pointer. Upon completion of the D command, EDIT positions the pointer immediately before the first character following the deleted text. Table 5-9 lists each argument for the D command and its effect.

Table 5-9 Delete Command Arguments

Argument	Meaning
n	Deletes n characters following the pointer. Places the pointer before the first character following the deleted text.
-n	Deletes n characters preceding the pointer. Places the pointer before the first character following the deleted text.
0	Deletes the current line up to the pointer. The position of the pointer does not change (equivalent to 0K).
/	Deletes the text between the pointer and the end of the buffer. Positions the pointer at the end of the buffer (equivalent to /K).
=	Deletes -n characters, where n equals the length of the last text argument used.

The following examples illustrate the use of the D command.

```
*-2D$$
```

This command deletes the two characters immediately preceding the pointer.

```
*B$FMOV R1$=D$$
```

This command string deletes the text string MOV R1. (=D in combination with a search command deletes the indicated text string.)

Assume the text buffer contains the following:

```
ADD    R1,(R2)+
CLR    ↑@R2
```

The following command deletes the current line up to the pointer:

```
*0D$$
```

The buffer now contains:

```
ADD    R1,(R2)+
↑@R2
```

5.6.6.3 Kill – The Kill (nK) command removes n lines of text (including the carriage return and line feed characters) from the page buffer, beginning at the pointer and ending with the nth end-of-line. The syntax of the command is:

```
nK
```

EDIT places the pointer at the beginning of the line following the deleted text. Table 5-10 describes each argument and its effect upon the Kill command.

Table 5-10 Kill Command Arguments

Argument	Meaning
n	Removes the character string (including the carriage return/line feed combination) beginning at the pointer and ending at the nth end-of-line.
-n	Removes the character string beginning at the nth end-of-line preceding the pointer and ending at the pointer. Thus, if the pointer is at the center of a line, the modifier -1 deletes one and one-half lines preceding it.
0	Removes the current line up to the pointer (equivalent to 0D).
/	Removes the characters beginning at the pointer and ending with the last line in the text buffer (equivalent to /D).

The following examples use the K command.

```
*2K$$
```

This command deletes lines starting at the current location of the pointer and ending at the second carriage return/ line feed combination.

Assume the text buffer contains the following:

```
ADD   R1,(R2)+
CLR   @R2
MOVB  6(R1),@R2
```

This command removes the characters beginning at the pointer and ending with the last line in the text buffer:

```
*/K$$
```

The buffer now contains:

```
ADD   R1,(R2)+
CLR
```

Kill and Delete commands perform the same function, except that Kill is line-oriented and Delete is character-oriented.

5.6.6.4 Change – The Change (nC) command changes a specific number of characters following the pointer. The syntax of the command is:

```
nCtext
```

A C command is equivalent to a Delete command followed by an Insert command. You must insert a text object following the nC command. Table 5-11 lists each argument and its effect upon the C command.

Table 5-11 Change Command Arguments

Argument	Meaning
n	Replaces n characters following the pointer with the specified text. Places the pointer after the inserted text.
-n	Replaces n characters preceding the pointer with the specified text. Places the pointer after the inserted text.
0	Replaces the current line up to the pointer with the specified text. Places the pointer after the inserted text (equivalent to 0X).
/	Replaces the text beginning at the pointer and ending with the last character in the buffer. Places the pointer after the inserted text (equivalent to /X).
=	Replaces -n characters with the indicated text string, where n represents the length of the last text argument used.

The size of the text is limited only by the size of the text buffer and the space available. All characters are legal except ESCAPE, which terminates the text string.

If the C command is to be executed more than once (i.e., it is enclosed in angle brackets) and if there is enough space available for the command to be entered, it will be executed at least once (provided it appears first in the command string). If repetition of the command exceeds the available memory, an error message prints.

The following examples use the C command.

```
*5C#VECT$$
```

This command replaces the five characters to the right of the pointer with #VECT.

Assume the text buffer contains the following:

```
CLR    @R2
MOV    5(R1),@R2
```

The next command replaces the current line up to the pointer with the specified text.

```
*0CADDB$$
```

The buffer now contains:

```
CLR    @R2
ADDB   5(R1),@R2
```

You can use =C with a Get command to replace a specific text string. Here is an example:

```
*GFIFTY:$=CFIVE:$
```

This command finds the occurrence of the text string FIFTY: and replaces it with the text string FIVE:.

5.6.6.5 eXchange – The eXchange (nX) command is similar to the change command except that it changes lines of text, instead of a specific number of characters. The syntax of the command is:

nXtext

The nX command is identical to an nK command followed by an Insert command. Table 5-12 lists each argument and its effect upon the eXchange command.

Table 5-12 eXchange Command Arguments

Argument	Meaning
n	Replaces n lines including the carriage return and line feed characters following the pointer. Places the pointer after the inserted text.
-n	Replaces n lines including the carriage return and line feed characters preceding the pointer. Positions the pointer after the inserted text.
0	Replaces the current line up to the pointer with the specified text. Positions the pointer after the inserted text (equivalent to 0C).
/	Replaces the text beginning at the pointer and ending with the last character in the buffer with the specified text (equivalent to /C). Positions the pointer after the inserted text.

All characters are legal in the text string except ESCAPE, which terminates the text.

If the X command is enclosed within angle brackets to allow more than one execution, and if there is enough memory space available for the X command to be entered, EDIT executes it at least once (provided it is first in the command string). If repetition of the command exceeds the available memory, an error message prints.

The following example uses the X command.

```
*2XADD  R1,(R2)+
 CLR    @R2
 $$
*
```

This command exchanges the two lines to the right of the pointer with the text string.

5.6.7 Utility Commands

During the editing session, you can store text in external buffers and subsequently restore this text when you need it later on in the editing session. The following sections describe the commands that perform this function.

5.6.7.1 Save – The Save (nS) command lets you store text in an external buffer called a save buffer (described previously in Section 5.5), and subsequently insert it in several places in the text.

The syntax of the command is:

nS

The Save command copies n lines, beginning at the pointer, into the save buffer. The S command operates only in the forward direction; therefore, you cannot use a negative argument. The Save command destroys any previous contents of the save buffer; however, EDIT does not change the location of the pointer or the contents of the text buffer.

If you specify more characters than the save buffer can hold, EDIT prints ?EDIT-F-INSUFFICIENT MEMORY. None of the specified text is saved.

For example, assume the text buffer contains the following assembly-language subroutine:

```
;SUBROUTINE MSGTYP
;WHEN CALLED, EXPECTS R0 TO POINT TO AN
;ASCII MESSAGE THAT ENDS IN A ZERO BYTE,
;TYPES THAT MESSAGE ON THE USER TERMINAL

MSGTYP:   TSTB    (R0)          ;DONE?
          BEQ     MDONE         ;YES-RETURN
MLOOP:    TSTB    @#177564      ;NO-IS TERMINAL READY?
          BPL     MLOOP         ;NO-WAIT
          MOVB    (R0)+,@#177566 ;YES PRINT CHARACTER
          BR      MSGTYP        ;LOOP
MDONE:    RTS     PC            ;RETURN
```

The following command stores the entire subroutine in the save buffer (assuming the pointer is at the beginning of the buffer):

```
*12S$$
```

You can insert the contents of the save buffer into a program whenever you choose by using the Unsave command.

5.6.7.2 Unsave – The Unsave (U) command inserts the entire contents of the save buffer into the text buffer at the pointer and leaves the pointer positioned following the inserted text. You can use the U command to move blocks of text or to insert the same block of text in several places. Table 5-13 lists the U commands and their meanings.

Table 5-13 U Command and Arguments

Command	Meaning
U	Inserts the contents of the save buffer into the text buffer.
0U	Clears the save buffer and reclaims the area for text.

The only argument the U command accepts is 0.

The contents of the save buffer are not destroyed by the Unsave command (only by the 0U command) and can be unsaved as many times as desired. If the Unsave command causes an overflow of the text buffer, the ?EDIT-F-INSUFFICIENT MEMORY error message prints, and the command does not execute.

For example:

```
*U$$
```

This command inserts the contents of the save buffer into the text buffer.

5.6.7.3 Macro – The Macro (M) command inserts a command string into the EDIT macro buffer. Table 5-14 lists the M commands and their meanings.

Table 5-14 M Command and Arguments

Command	Meaning
M/command string/	Stores the command string in the macro buffer.
0M or M//	Clears the macro buffer and reclaims the area for text.

The slash (/) represents the delimiter character. The delimiter is always the first character following the M command, and can be any character that does not appear in the macro command string itself.

Starting with the character following the delimiter, EDIT places the macro command string characters into its internal macro buffer until the delimiter is encountered again. At this point, EDIT returns to command mode. The macro command does not execute the macro string; it merely stores the command string so that the Execute Macro (EM) command can execute later. The Macro command does not affect the contents of the text or save buffers.

All characters except the delimiter are legal macro command string characters, including single ESCAPEs to terminate text commands. All commands, except the M and EM commands, are legal in a command string macro.

In addition to using the 0M command, you can type the M command immediately followed by two identical characters (assumed to be delimiters) and two ESCAPE characters to clear the macro buffer.

The following examples illustrate the use of the M command.

```
*M//$$
```

This command clears the macro buffer.

```
*M/GR0$-C1$/$$
```

This command stores a macro to change R0 to R1.

NOTE

Be careful to choose infrequently-used characters as macro delimiters; use of frequently-used characters can lead to inadvertent errors. For example:

```
*M GMOV R0$=CADD R1$ $$
?EDIT-F-No file open for input
```

In this case, it was intended that the macro be GMOV R0$=CADD R1$ but since the delimiter character (the character following the M) is a space, the space following MOV is used as the second delimiter, terminating the macro. EDIT then returns an error when it interprets the R as a Read command.

5.6.7.4 Execute Macro – The Execute Macro (nEM) command executes the command string previously stored in the macro buffer by the M command.

The syntax of the command is:

nEM

The argument (n) must be positive. The macro is executed n times and returns control to the next command in the original command string.

The following example uses the EM command.

```
*M/BGR0$-C1$/$$
*B1000EM$$
?EDIT-F-Search failed
*
```

This command sequence executes the macro stored in the previous example. EDIT prints an error message when it reaches the end of the buffer. (This macro changes all occurrences of R0 in the text buffer to R1.)

```
*IMOV PC,R1$2EMICLR @R2$$
*
```

This command inserts MOV PC,R1 into the text buffer, then executes the command in the macro buffer twice before inserting CLR @R2 into the text buffer.

5.6.7.5 Edit Version – The Edit Version (EV) command displays the version number of the editor in use on the console terminal.

The command is:

```
EV
```

This example displays the running version of EDIT:

```
*EV$$
V03.36
*
```

5.6.7.6 Upper- and Lower-Case Commands – If you have an upper- and lower-case terminal as part of your hardware configuration, you can take advantage of the upper- and lower-case capability of this terminal. Two editing commands, EL and EU, permit this.

When the editor is first started with the EDIT command, upper-case mode is assumed; all characters you type are automatically translated to upper case. To allow processing of both upper- and lower-case characters, enter the Edit Lower command. For example:

```
*EL$$
*i You can enter text and commands in UPPER and lower case.$$
*
```

The editor now accepts and echoes upper- and lower-case characters received from the keyboard and prints text on the terminal in upper and lower case.

To return to upper-case mode, use the Edit Upper command:

```
*EU$$
```

Control also reverts to upper-case mode upon exit from the editor (with EX or CTRL/C).

Note that when you issue an EL command, you can enter EDIT commands in either upper or lower case. Thus, the following two commands are equivalent:

```
*GTEXT$=Cnew text$V$$

*gTEXT$=cnew text$v$$
```

The editor automatically translates (internally) all commands to upper case independent of EL or EU.

NOTE

When you use EDIT in EL mode, make sure that text arguments you specify in search commands have the proper case. The command GTeXt$, for example, will not match TEXT, text, or any combination other than TeXt.

5.7 THE DISPLAY EDITOR

In addition to all functions and commands mentioned thus far, the editor can use VT-11 and VS-60 display hardware that may be part of the system configuration (GT40, GT44, DECLAB 11/40, DECLAB 11/34). The most obvious feature is the ability to use the display screen rather than the console terminal for printing all terminal input and output. Another feature is that the top of the display screen functions like a window into the text buffer. When all the features of the display editor are in use, a 12 in. screen displays text as shown in Figure 5-1.

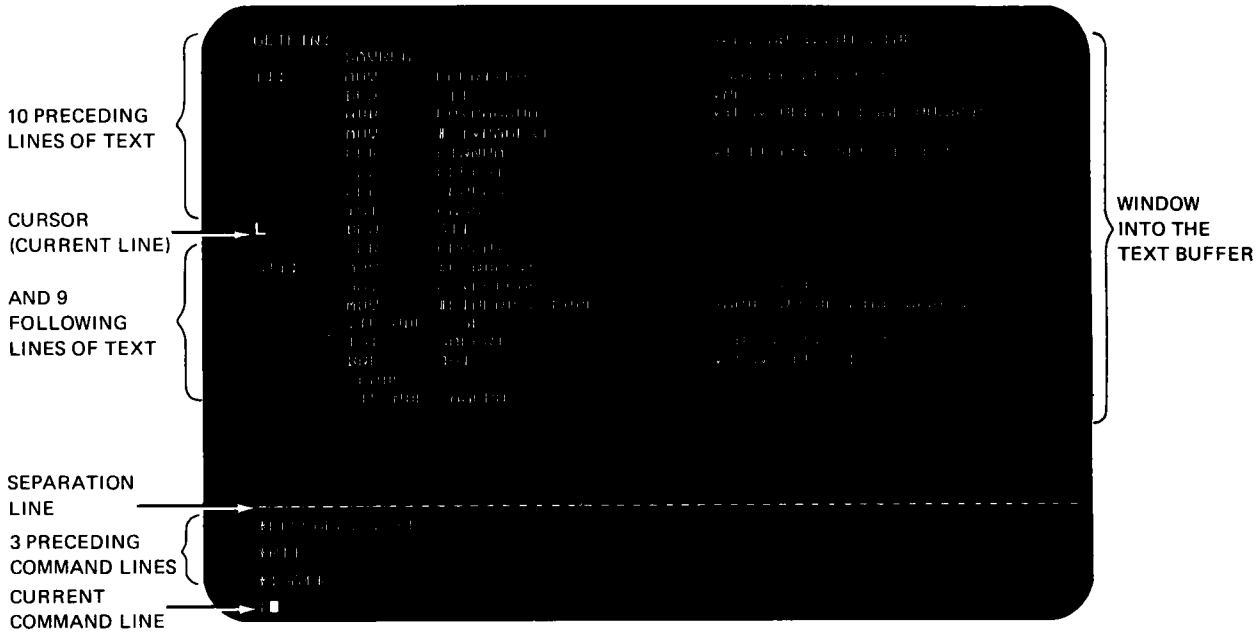

10 PRECEDING LINES OF TEXT
CURSOR (CURRENT LINE)
AND 9 FOLLOWING LINES OF TEXT
WINDOW INTO THE TEXT BUFFER
SEPARATION LINE
3 PRECEDING COMMAND LINES
CURRENT COMMAND LINE

Figure 5-1 Display Editor Format, 12 in. Screen

The major advantage is that you can now see immediately where the pointer is. The pointer appears between characters on the screen as a blinking L-shaped cursor and you can see it easily. Remember that pressing the (RET) key causes both a carriage return and a line feed character to be inserted into the text. Note that if the pointer is placed between a carriage return and line feed, it appears in an inverted position at the beginning of the next line.

In addition to displaying the current line (the line containing the cursor), the 15 lines of text preceding the current line and the 14 lines following it are also in view on a 17 in. screen. Each time you execute a command string (with a double ESCAPE), EDIT refreshes this portion of the screen so that it reflects the results of the commands you just performed.

The lower section of the 17 in. screen contains eight lines of editing commands. The command line you are currently entering is last, preceded by the most recent command lines. A horizontal line of dashes separates this section from the text portion of the screen. As you enter new command lines, previous command lines scroll upward off the command section so that only eight command lines are ever in view.

A 12 in. screen displays 20 lines of text and 4 command lines.

5.7.1 Using the Display Editor

The display features of the editor are automatically invoked whenever the system scroller is in use (a monitor GT ON command is in effect) and you start the editor. However, if the system does not contain display hardware, the display features are not enabled.

Providing that the system does contain display hardware and that you wish to employ the screen during the editing session, you can activate it in one of two ways, whether or not the display is in use. All editing commands and functions previously discussed in this chapter are valid for use.

1. If the scroller is in use (the GT ON monitor command is in effect), EDIT recognizes this and automatically uses the screen for display of text and commands. However, it rearranges the scroller so that a window into the text buffer appears in the top two-thirds of the screen, while the bottom third displays command lines. This arrangement is shown in Figure 5-1.

   You can use the Edit Console command to return the scroller to its normal mode so that text and commands use the full screen, and the window is eliminated.

   The command is:

   ```
   EC
   ```

   This example uses the EC command:

   ```
   *BAEC2L$$
   ```

   This command lists the second and third lines of the current buffer on the screen; there is no window into the text buffer at this point.

   EDIT ignores subsequent EC commands if the window into the text buffer is not being displayed.

   To recall the window, use the Edit Display command:

   ```
   ED
   ```

   The screen is again arranged as shown in Figure 5-1.

2. Assume the scroller is not is use (the GT ON command is not in effect). When you call EDIT with the .EDIT command, an asterisk appears on the console terminal. Use the ED command at this time to provide the window into the text buffer; however, commands continue to be echoed to the console terminal.

   When you use ED in this case, it must be the first command you issue; otherwise, it becomes an illegal command (since the memory used by the display buffer and code, amounting to over 600 words, is reclaimed as working space). You cannot use the display again until you load a fresh copy of EDIT.

   While the display of the text window is active, EDIT ignores ED commands.

   Typing the EC command clears the screen and returns all output to the console terminal.

NOTE

After an editing session that uses the ED command is over, clear the screen by typing the EC command or by returning to the monitor and using the monitor RESET command. Failure to do this may cause unpredictable results.

5.7.2 Setting the Editor to Immediate Mode

An additional mode is available in EDIT to provide easier and faster interaction during the editing session. This mode is called immediate mode and combines the most-used functions of the text and command modes – namely, repositioning the pointer and deleting and inserting characters.

You can only use immediate mode when the VT-11 display hardware is active and the editor is running. Enter it by typing two ESCAPEs (only) in response to the command mode asterisk:

```
*$$
```

The editor responds by echoing an exclamation point on the screen.

```
!
```

The exclamation character remains on the screen as long as control is in immediate mode.

Once you enter immediate mode, you can use only the commands in Table 5-15. Any other commands or characters are treated as text to be inserted. None of these commands echoes, but the text appearing on the screen is constantly refreshed and updated during the editing process.

To return control to the display editor's normal command mode at any time while in immediate mode, type a single ESCAPE. The editor responds with an asterisk and you can proceed using all normal editing commands. (Immediate mode commands you type at this time will be accepted as command mode input characters.) To return control to the monitor while in immediate mode, type ESCAPE to return to command mode, then type CTRL/C followed by two ESCAPEs.

Table 5-15 Immediate Mode Commands

Command	Meaning
CTRL/N	Advances the pointer (cursor) to the beginning of the next line (equivalent to A).
CTRL/G	Moves the pointer (cursor) to the beginning of the previous line (equivalent to -A).

(Continued on next page)

Table 5-15 (Cont.) Immediate Mode Commands

Command	Meaning
CTRL/D	Moves the pointer (cursor) forward by one character (equivalent to J).
CTRL/V	Moves the pointer (cursor) back by one character (equivalent to -J).
RUBOUT or DELETE	Deletes the character immediately preceding the pointer (cursor) (equivalent to -D).
ESCAPE or ALTMODE	Single character returns control to command mode; double character directs control to immediate mode.
Any character other than those above	Inserts the character as text positioned immediately before the pointer (cursor) – equivalent to I.

5.8 EDIT EXAMPLE

The following example illustrates the use of some of the EDIT commands to change a program stored on the device DK:. Sections of the terminal output are coded by letter and corresponding explanations follow the example.

```
A  EDIT/OUTPUT:TEST2.MAC TEST1.MAC
   *R$$
B  *0/L$$
   ;TEST PROGRAM

   START:  MOV     #1000,SP        ;INITIALIZE STACK
           MOV     #MSG,R0         ;POINT R0 TO MESSAGE
           JSR     PC,MSGTYP       ;PRINT IT
           HALT                    ;STOP
   MSG:    .ASCII/IT WORKS/
           .BYTE 15
           .BYTE 12
           .BYTE 0
C  *B$1J$5D$$
D  *GPROGRAM$$
   *0L$$
E  ;PROGRAM*I TO TEST SUBROUTINE MSGTYP. TYPES
   ;"THE TEST PROGRAM WORKS"
   ;ON THE TEMI\IM\RMINAL$$
F   G.ASCII/$$
   *8CTHE TEST PROGRAM WORKS$$
G  *P.BYTE^X
   *G.BYTE 0$V$$
           .BYTE 0
```

```
*I
          .END
$B/L$$
;PROGRAM TO TEST SUBROUTINE MSGTYP. TYPES
;"THE TEST PROGRAM WORKS"
;ON THE TERMINAL

START:    MOV       #1000,SP          ;INITIALIZE STACK
          MOV       #MSG,R0           ;POINT R0 TO MESSAGE
          JSR       PC,MSGTYP         ;PRINT IT
          HALT                        ;STOP
MSG:      .ASCII/THE TEST PROGRAM WORKS/
          .BYTE 15
          .BYTE 12
          .BYTE 0
          .END


*EX$$

.
```

A Calls the EDIT program and prints *. The input file is TEST1.MAC; the output file is TEST2.MAC. Reads the first page of input into the buffer.

B Lists the buffer contents.

C Places the pointer at the beginning of the buffer. Advances the pointer one character (past the ;) and deletes the TEST.

D Positions the pointer after PROGRAM and verifies the position by listing up to the pointer.

E Inserts text. Uses RUBOUT to correct typing error.

F Searches for .ASCII/ and changes IT WORKS to THE TEST PROGRAM WORKS.

G Types CTRL/X to cancel the P command. Searches for .BYTE 0 and verifies the location of the pointer with the V command.

H Inserts text. Returns the pointer to the beginning of the buffer and lists the entire contents of the buffer.

I Closes the input and output files after copying the current text buffer as well as the rest of the input file into the output file. EDIT returns control to the monitor.

5.9 EDIT ERROR CONDITIONS

The editor prints an error message whenever a detectable error condition occurs. EDIT checks for three general types of error conditions: 1) syntax errors, 2) execution errors and, 3) macro execution errors. This section describes the error message form for each type of error condition.

Before it executes any commands, EDIT first scans the entire command string for errors in command syntax, such as illegal arguments or an illegal combination of commands. If the editor finds an error of this type, it prints a message of this form:

?EDIT-F-Message; no command(s) executed

You should retype the command.

If a command string is syntactically correct, EDIT begins execution. Execution errors, such as buffer overflow or input and output errors, can still occur. In this case, EDIT prints a message of the form:

```
?EDIT-F-Message
```

EDIT executes all commands preceding the one in error. It does not execute the command in error or any commands that follow it.

When an error occurs during execution of a macro, EDIT prints a message of the form:

```
?EDIT-F-Message in macro; no command(s) executed
or
?EDIT-F-Message in macro
```

Most errors are syntax errors. These are usually easy to correct before execution.

The *RT-11 System Message Manual* contains a complete list of the EDIT error messages, along with recommended corrective action for each error.

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. Problems with software should be reported on a Software Performance Report (SPR) form. If you require a written reply and are eligible to receive one under SPR service, submit your comments on an SPR form.

Did you find errors in this manual? If so, specify by page.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Is there sufficient documentation on associated system programs required for use of the software described in this manual? If not, what material is missing and where should it be placed?

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Non-programmer interested in computer concepts and capabilities

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

Please cut along this line.

---------- Fold Here ----------

---------- Do Not Tear - Fold Here and Staple ----------

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Software Documentation
146 Main Street ML5-5/E39
Maynard, Massachusetts 01754